SISTEMAS INFORMÁTICOS 2010/2011

DES GUI Front-end

Realizado por:

Javier Salcedo Gómez

Dirigido por:

Prof. Fernando Sáenz Pérez

Dpto. Ingeniería del Software e Inteligencia Artificial

FACULTAD DE INFORMÁTICA

UNIVERSIDAD COMPLUTENSE DE MADRID

ÍNDICE DE CONTENIDOS

1 RESUMEN DEL PROYECTO

Este proyecto consiste en la implementación de una nueva versión de un proyecto anterior llamado "ACIDE: A Configurable IDE" realizado durante el curso académico 2006-2007 por los alumnos Diego Cardiel Freire, Juan José Ortiz Sánchez y Delfín Rupérez Cañas en primer lugar, y posteriormente por el alumno Miguel Martín Lázaro durante el curso académico 2007-2008, siempre dirigido por Fernando Sáenz Pérez.

ACIDE es un entorno de desarrollo integrado y configurable para diferentes lenguajes de programación. Los detalles de dicho proyecto pueden ser consultados en la memoria [1] referenciada en la Bibliografía.

La anterior versión del proyecto presentaba numerosas funcionalidades útiles y eficaces, una gestión de proyectos sólida y un comportamiento fiable en líneas generales.

No obstante, dicha aplicación carecía de un aspecto básico para un software cuyo código pretende ser de libre distribución, a saber, un código fuente no estandarizado, bastante complejo y descuidado en cuanto a su presentación e implementación.

Además de haber hecho especial hincapié en este aspecto, también se han adoptado medidas para hacer que la aplicación sea un poco más amigable en el aspecto gráfico y una serie de nuevas funcionalidades que convierten la aplicación en una aplicación más completa y funcional que las anteriores.

2 ABSTRACT

This project consists of the implementation of a new version of a previous project called "ACIDE: A Configurable IDE" made by the students Diego Cardiel Freire, Juan José Ortiz Sánchez y Delfín Rupérez Cañas during the 2006-2007 academical year, and by the student Miguel Martín Lázaro during the 2007-2008 academic year, supervised by Fernando Sáenz Pérez in both cases.

ACIDE is an integrated development environment which can be configured for different programming languages. The details about that project can be consulted in the written paper [1] referenced in the bibliography.

The previous version of the project, presented numerous useful and handy functionalities but it was lacking of a basic and desirable feature of an open source applicaction: a messy and non-standard source code with other problems that will be further discussed in the following chapters of the present document.

Besides of having put a lot of emphasis on this aspect, some graphical and new interesting functionalities has been added to the project in order to improve and make it more friendly and complete to the final user.

3 ESTADO DEL ARTE

Para conocer el estado del arte en el momento del inicio de la creación de ACIDE así como en la primera revisión del mismo, se remite al autor a la lectura del apartado con el mismo nombre que el presente en la publicación correspondiente [1].

Durante el período acaecido entre esta primera revisión y el proyecto actual, han sido numerosos proyectos con características similares a ACIDE los que han ido apareciendo o evolucionando hacia versiones más completas y funcionales como por ejemplo **Microsoft Visual Studio**, los propios **Eclipse** y **NetBeans** usados para el desarrollo del presente proyecto o los mencionados **JEdit**, **WinEdt** y **CrimsonEditor**.

No obstante, se ha seguido tomando como referencia **JEdit** [2], **WinEdt** [3] y **CrimsonEditor** [4] principalmente y, en menor medida, **Eclipse** [5] y **NetBeans** [6] para la extracción de nuevas funcionalidades a añadir o mejorar en el proyecto.

4 ESTÁNDARES

La importancia de la aplicación de estándares sobre el desarrollo de un proyecto de gran envergadura es totalmente deseable y necesaria, especialmente cuando se trata de una aplicación de código abierto o cuando en ella parcipan un número considerable de personas.

Nótese que se han tenido en cuenta los anteriores estándares aplicados en las versiones previas del proyecto. No obstante, se han modificado alguno de ellos para la aplicación de las medidas necesarias para las mejoras del mismo y que son detalladas a continuación:

4.1 CONTROL DE VERSIONES

Para el control de versiones se ha utilizado el cliente **subversion** [7] y el repositorio gratuíto de google **"google code"** [8].

Con cada nueva versión de la aplicación, se ha subido a dicho repositorio un archivo RAR que contiene la versión ejecutable del mismo en su interior. Por convenio, se ha aplicado una contraseña para que solamente el alumno y supervisor tuvieran acceso a las versiones experimentales del software en dichos momentos. Durante las primeras versiones ejecutables del mismo no se adoptó convenio alguno para la nomenclatura de dichos archivos. Posteriormente, y ante la incapacidad del repositorio para guardar archivos con el mismo nombre se decidió llamar a los archivos ejecutables con la siguiente nomenclatura: **"ACIDE0.8_-_día_mes_año.rar"** para identificar inequívocamente a cada uno de ellos en el repositorio. Además se tomaron las siguientes medidas para cada una de las versiones de las descargas:

- El campo **Summary** contiene el texto **"IDE ACIDE 0.8 día-mes-año".**
- El campo **Description** contiene una descripción en inglés con las funcionalidades añadidas, modificadas y eliminadas en dicha versión.
- Como información adicional se han añadido las siguientes etiquetas: **"Type-Executable"** y **"OpSys-All"**

El repositorio de datos contiene la siguiente estructura:

- **svn:** directorio principal del repositorio.
  - **branches:** contiene las versiones importantes que se han comportado de forma estable durante el desarrollo del proyecto.
  - **tags:** contiene la documentación del proyecto, en este caso todas los documentos de lista de tareas que se han ido elaborando durante el desarrollo del proyecto.
  - **trunk:** contiene el código fuente del proyecto.
  - **wiki:** dedicado a documentación acerca del proyecto, pero dado que con los documentos sobre listas de tareas fue suficiente para la comunicación entre alumno y profesor, no ha sido usado finalmente.

4.2 DOCUMENTACIÓN

Para la comunicación directa entre alumno y profesor durante el desarrollo del proyecto, se ha procedido a la redacción periódica de una serie de documentos de seguimiento del mismo. En dichos documentos, se podía encontrar el estado actual del proyecto con la lista de *tareas realizadas* o *pendientes de realización* y se modificaban para su posterior envío junto a un nuevo ejecutable de la aplicación.

Para identificar dichos documentos y llevar un control de los mismos, se ha seguido la siguiente nomenclatura para los mismos: **"Año-Mes-Día_Lista_de_tareas_ACIDE.doc"**.

Con respecto a su contenido, éste se ha dividido en dos secciones principales: *Tareas Realizadas* y *Tareas Pendientes*. Cada una de estas categorías se ha dividido, a su vez, en otras categorías diferentes relacionadas cada una de ellas con un aspecto diferente de la aplicación incluyendo *tareas urgentes* y *futuras funcionalidades*.

Para una mejor comprensión de dichos documentos, se ha procedido a la definición de una leyenda para explicar los diferentes colores de las tareas en los mismos:

- Tareas validadas.
- Tareas pendientes.
- Tareas implementadas pero no validadas.
- Aclaraciones del alumno.
- Aclaraciones del supervisor.

En dichos documentos se ha aplicado el siguiente estándar:

- El estilo del texto para todo el documento se compone de fuente Arial, tamaño 12pt, justificado, sangría de 0,7cm en la primera línea, espacio anterior y posterior de párrafo de 6pt y color negro.
- El estilo para la cabecera de primer nivel se compone de fuente Calibri, tamaño 16pt, justificado, letras versales con sombra, sangría de 0,7cm en la primera línea, espacio anterior de 24pt y posterior de 15pt y color azul marino.
- El estilo para la cabecera de primer nivel se compone de fuente Calibri, tamaño 14pt, justificado, letras versales con sombra, sangría de 0,7cm en la primera línea, espacio anterior de 24pt y posterior de 15pt y color azul claro.
- Pie de página con texto *"2007-2011©"* seguido de la numeración de página en pie de página con formato *"página x de y"*. El formato del mismo tiene fuente Cambria, tamaño 11, justificado con borde superior azul marino.
- Encabezado con las imágenes del logotipo de la aplicación, el de la Facultad de Informática y el logotipo de la Universidad complutense de Madrid respectivamente.
- Para las listas de enumeraciones se ha usado la herramienta de Microsoft Word.

Con respecto al estándar del presente documento del manual de usuario:

- El estilo del texto para todo el documento se compone de fuente Cambria, tamaño 12pt, justificado, interlineado de 1,5pt, sangría de 0,7cm en la primera línea, espacio anterior y posterior de párrafo de 6pt y color negro.
- El estilo para la cabecera de primer nivel se compone de fuente Cambria, tamaño 16pt, justificado, letras versales, interlineado de 1,5pt, sangría de 0,7cm en la primera línea, espacio anterior de 24pt y posterior de 15pt y color azul marino.
- El estilo para la cabecera de segundo nivel se compone de fuente Cambria, tamaño 14pt, justificado, letras versales, interlineado de 1,5pt, sangría de 0,7cm en la primera línea, espacio anterior de 24pt y posterior de 15pt y color azul marino.
- Para el formato de código fuente se ha utilizado una fuente Courrier, un tamaño de 11pt, alineación a la izquierda y un borde de color negro de 1pt.
- Pie de página con texto *"Sistemas Informáticos 2010-2011"* seguido de la numeración de página en pie de página con formato *"página x de y"*. El formato del mismo tiene fuente Cambria, tamaño 11, justificado con borde superior azul marino de 1pt.
- Encabezado con las imágenes del logotipo de la aplicación, el de la Facultad de Informática y el logotipo de la Universidad complutense de Madrid respectivamente.
- Para las listas de enumeraciones se usará la herramienta de Microsoft Word.

4.3 CÓDIGO FUENTE

Para la estandarización del código fuente de la aplicación se han llevado a cabo las siguientes medidas:

- Traducción del mismo al idioma **Inglés**.
- Código de licencia pública **GPLv3** disponible en cada una de las diferentes clases del código fuente al principio de las mismas:

```
/*
 * ACIDE - A Configurable IDE
 * Official web site: http://acide.sourceforge.net
 *
 * Copyright (C) 2007-2011
 *
 * Authors:
 *   - Fernando Sáenz Pérez (Team Director).
 *   - Version from 0.1 to 0.6:
 *       - Diego Cardiel Freire.
 *       - Juan José Ortiz Sánchez.
 *         - Delfín Rupérez Cañas.
 *   - Version 0.7:
 *        - Miguel Martín Lázaro.
 *   - Version 0.8:
 *        - Javier Salcedo Gómez.
 *
 * This program is free software: you can redistribute
 * it and/or modify it under the terms of the GNU General Public
 * License as published by the Free Software Foundation,
 * either version 3 of the License, or (at your option) any
 * later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public
 * License along with this program.  If not, see
```

```
 * <http://www.gnu.org/licenses/>.
 */
```

- Comentarios **Javadoc**, simples y multilínea introducidos en cada una de las líneas de código de la aplicación.

```
// Gets the selected file editor panel index
int selectedFileEditorPanelIndex =
AcideMainWindow.getInstance().getFileEditorManager().getSelectedFil
eEditorPanelIndex();

/*
 * IMPORTANT: This is mandatory to avoid exceptions during the
 * workbench loading process.
 */
if(AcideWorkbenchConfiguration().getInstance().isLoaded()){...}
```

- Para los comentarios **Javadoc**, por cada clase del código se tendrá el formato siguiente:

```
/**
 * Descripción de la clase.
 *
 * @version 0.8
 * (@see <NombreDeClase/NombreDeInterfaz>)
 */
```

- Variables de clase precedidas por "_":

```
private JButton _acceptButton;
private JColor _backgroundColor;
```

- Nombres de clase comienzan por "**Acide**" seguidos por palabras cuya primera letra es mayúscula:

```
public class AcideResourceManager{...}
```

- Nombres de métodos comienzan por minúscula y si tienen más de una palabra, cada una de ellas comenzará con mayúsculas.

```
public AcideFileEditorPanel getSelectedFileEditor(){...}
```

- Para las constantes, todas las letras en mayúsculas y separadas por "_":

```
public static final String DEFAULT_PATH;
```

- En los menús, las constantes para los nombres de las diferentes opciones en cuanto a su nombre terminan con "**NAME**" y si son para su icono lo hacen con "**IMAGE**":

```
private String SHOW_CONSOLE_PANEL_NAME;
private ImageIcon SHOW_CONSOLE_PANEL_IMAGE;
```

- En las clases correspondientes a ventanas de configuración, todos los nombres de variables contienen al final el tipo de componente del que se trata:

```
private JMenuItem _showConsoleMenuItem;
private JTextField _shellPathTextField;
private JLabel _shellPathLabel;
```

- Todas las clases correspondientes a ventanas de configuración contienen los siguientes métodos:

```
// Crea y configura los componentes de la ventana
private void initComponents(){...}
// Añade los componentes a la ventana
private void addComponents(){...}
// Establece el título, tamaño, muestra la ventana, etc.
private void setWindowConfiguration{...}
// Establece los listeners para los componentes de la ventana.
private void setListeners(){...}
// Cierra la ventana.
private void closeWindow(){...}
```

- Todas las clases de la barra de menús y menús contextuales tienen los siguientes métodos:

```
// Crea y configura los componentes del menú
private void buildComponents(){...}
// Añade los componentes al menú
private void addComponents(){...}
// Establece el texto y los atajos de las opciones del menú.
private void setTextOfMenuComponents{...}
```

```
// Establece la visibilidad o invisibilidad de los componentes del
menú.
private void updateComponentsVisibility(){...}
// Establece los listeners para los componentes de la ventana.
private void setListeners(){...}
```

5 GESTIÓN DE LA CONFIGURACIÓN

Se ha respetado la gestión de la configuración descrita en [1].

Para la realización del proyecto se ha utilizado el siguiente software:

- **Eclipse SDK versión 3.6.2** para el desarrollo del código fuente.
- **Microsoft Office 2007** para la documentación del proyecto.
- **Tortoise SVN** para la interacción con el repositorio de datos.
- **WinRar** para la generación de los archivos comprimidos que contienen el ejecutable de la aplicación.
- **Adobe Photoshop CS** para la edición de los iconos, logotipo e imágenes del proyecto.

6 GESTIÓN DE REQUISITOS

Se ha respetado la gestión de requisitos inicial mencionada en [1], no obstante se ha ido modificando la misma durante el desarrollo del proyecto conforme iban surgiendo nuevas posibilidades.

Tras las primeras tomas de contacto antes del inicio del desarrollo del proyecto los principales requisitos fueron los siguientes:

- Aumento de funcionalidades.
- Eliminación de errores existentes.

Otro requisito inicial fue una mayor integración de la herramienta **DES** [9] en la aplicación.

No obstante, con el inicio del desarrollo del proyecto surgió un requisito que ha supuesto a posteriori el mayor esfuerzo durante el desarrollo del mismo: **estandarización y optimización del código fuente**.

Otro de los requisitos fundamentales que surgieron a simple vista fue el aspecto gráfico un tanto descuidado y poco amigable que tenía la aplicación. Cuando el usuario iniciaba la aplicación, se mostraba al usuario una ventana un tanto simple al usuario informando de la carga de la aplicación. Una vez lanzada, la barra de herramientas presentaba unos iconos bastante simples y los menús, tanto de la barra como los menús contextuales no tenían iconos. Por tanto se decidió incorporar iconos a todos los menús, cambiar los ya existentes y añadir un **SplashScreen** mostrando una imagen con el logotipo de la aplicación, para dotar a la misma de un aspecto más profesional.

7 PLANIFICACIÓN

Para la realización del proyecto no se ha llevado a cabo ninguna tarea de planificación del mismo de una manera formal.

De una forma periódica, y siempre que se ha tenido disponible una nueva versión de la aplicación, se han ido enviando dichas versiones junto con la documentación oportuna al supervisor.

La planificación por tanto no ha sido estimada en términos de recursos, tiempo y esfuerzo, dado que se han ido implementado conforme dichas tareas eran más o menos frecuentes dependiendo del momento del desarrollo del mismo. Tampoco fue tenida en cuenta desde un principio dado el carácter individual del desarrollo del mismo y se decidió dedicar los esfuerzos entera y exclusivamente al desarrollo del código fuente y de la documentación pertinente en su lugar.

No obstante, en términos de iteraciones importantes y distinguibles el alumno ha identificado las siguientes iteraciones:

7.1 PRIMERA ITERACIÓN

Esta primera iteración va desde el **inicio del desarrollo del proyecto** hasta el **13 de Diciembre de 2010**.

Durante dicha etapa el alumno se ha ido familiarizando con el sistema, ha ido realizando las primeras tareas de la lista de menor a mayor dificultad a medida que se ha ido analizando el código.

Alguna de las mejoras que se introdujeron fueron:

- Aplicación de iconos en los menús y ventanas de configuración de la aplicación.
- Diseño del logotipo de la aplicación para la *ventana de splash* y opción de *"Acerca de"*.
- Refactorización de la clase menú.
- Refactorización del editor de textos.

- Estandarización del código con los estándares de código mencionados con anterioridad en el presente documento.
- Corrección de errores pendientes de la última versión del proyecto.

Al final de la misma, el sistema presentaba un funcionalidad no del todo estable y con un profundo rediseño del código en el menú y editor de textos de la aplicación.

7.2 Segunda Iteración

La segunda iteración se identifica desde el día **13 de Diciembre de 2010** hasta el día **3 de Enero de 2011**.

Durante esta fase se intentó corregir la falta de estabilidad de la aplicación relativa a la gestión de proyectos, editor de archivos y panel de la consola.

Algunas de las mejoras introducidas durante ésta iteración han sido:

- Primer rediseño de la configuración de menús para hacerla independiente del número de opciones de la barra de menús.
- Modificación de la presentación del panel de la consola en cuanto a tamaño, estilo, color de fuente y color de fondo y su aplicación a la configuración de proyectos.
- Adición de botones de scroll en la barra de herramientas cuando las dimensiones de la misma sobrepasan las dimensiones de la ventana de la aplicación.
- Primera aproximación al concepto del gestor de espacio de trabajo encargado de la gestión del editor de texto así como otras características propias del programa y no relativas al proyecto.

En torno al 3 de Enero, se produjo un parón de aproximadamente de un mes en el desarrollo del proyecto debido a los compromisos escolares del alumno, correspondientes a los exámenes y entregas finales de prácticas para la universidad.

7.3 Tercera Iteración

Desde **principios del mes de Febrero de 2011** hasta el día **31 de Marzo de 2011,** se desarrollaron grandes avances en el desarrollo de la aplicación.

Dichos avances se centraron sobretodo en:

- Definición del gestor de espacio de trabajo y cómo debía interactuar éste con la gestión de los proyectos en la aplicación.
- Rediseño de la ventana de configuración de la barra de herramientas y de menús así como sus gestores de configuración.
- Separación de la barra de herramientas en diferentes secciones.
- Activación o desactivación de las opciones de la barra de herramientas basadas en las opciones de menú.
- Aplicación de configuraciones léxicas y sintácticas para cada archivo abierto en el editor de texto.
- Solución al problema del foco del editor activo en el editor de texto.

La aplicación se encontró en un punto muerto durante la presente iteración en cuanto a la introducción de nuevas funcionalidades, centrándose todos los esfuerzos en la resolución del problema del foco del editor activo en el editor de texto.

Finalmente, con la release del 31 de Marzo de 2011, se alcanzó el punto en el que la aplicación se comportaba de una forma más o menos estable aunque todavía con carencias que necesitaban ser resueltas para su posterior publicación.

7.4 Cuarta Iteración

Ésta última iteración ha ido desde la publicación de la anterior release semi estable del **31 de Marzo de 2011** hasta el **19 de Junio de 2010** que se corresponde a la entrega final con una **release estable** y **distribuíble**.

Durante toda ésta iteración se han centrado todos los esfuerzos en hacer que la aplicación se comportase de una forma fiable y segura. A su vez se han ido introduciendo nuevas mejoras en la misma.

Como principales tareas añadidas se han añadido:

- Definición del gestor de léxicos por defecto para editores de texto y para el panel de la consola así como su ventana de configuración.
- El *line wrapping* en los editores de texto y las opciones para activarlo o evitarlo en la barra de menús y menú contextual en el editor de archivos.
- Envío de texto desde el editor de texto activo al panel de la consola mediante opción en la barra de menús y menú contextual en el panel de la consola.
- Definición del número máximo de líneas a enviar del editor de textos al panel de la consola.
- Scroll línea a línea en los editores de texto y en el panel de la consola con control+rueda de ratón.
- Scroll línea a línea en los editores de texto y en el panel de la consola con rueda de ratón.
- Búsquedas en el panel de la consola mediante opción de menú y menú contextual en el panel de la consola.
- Definición del alcance de la aplicación de archivos compilables y principales en el editor cuando están abiertos pero no pertenecen al proyecto.
- Guardar el contenido del panel de la consola en un archivo mediante opción de menú y menú contextual en el panel de la consola.
- Corrección de activación/desactivación de las opciones de la barra de menús y simplificación de las mismas.
- Gestión de recuperación de errores en la configuración del espacio de trabajo cuando la configuración del mismo es errónea.
- Edición sobre la consola solamente disponible tras la marca de prompt de la misma.
- Definición de tamaño máximo del buffer para el panel de la consola por número de líneas y su aplicación a la gestión de configuración de proyectos.

- Detección de teclas de Lock en todo momento y su posterior actualización en la barra de estado de la aplicación.
- Rollback sobre versión anterior para adecuar la versión existente a la misma y así conseguir la definitiva versión final de la aplicación.

Desde el 1 de Junio de 2011 hasta el 9 de Junio de 2011 se ha procedido a la redacción del presente documento. Posteriormente desde el 10 de Junio de 2011 hasta 19 de Junio de 2011 se han dedicado exclusivamente todos los esfuerzos en arreglar los problemas con la gestión de proyectos y las búsquedas que provocaban el colapso de la aplicación así como la fusión de una versión anterior con la actual para garantizar el corrector funcionamiento del editor de archivos con los nuevos cambios introducidos.

Para las **búsquedas** se ha optado finalmente por realizar un *rollback* restaurando las clases originales de la anterior versión y aplicando los cambios introducidos para ésta nueva versión.

Nota: Durante todas las iteraciones explicadas con anterioridad no han sido mencionadas todas los errores intermedios que se han ido solventando. Se entiende por errores intermedios todos esos errores que han sido provocados por pruebas que ha ido realizando el alumno o todas aquellas que han sido provocadas al añadir las nuevas funcionalidades ó al refactorizar el código fuente del mismo.

8 TAREAS REALIZADAS

A continuación se detallan las tareas que se han ido realizando durante el proyecto en términos de código fuente. Por código fuente entendemos que no solamente se han hecho modificaciones sobre las mismas clases en sí, ya sea modificándolas, añadiéndolas o eliminándolas, sino también sobre los diferentes **paquetes**, **librerías**, **recursos** (imágenes, iconos y ayuda principalmente) y archivos de **configuración de ACIDE** , **XML** y ficheros de **properties** principalmente.

8.1 ESTANDARIZACIÓN DE LA APLICACIÓN

Como ya se ha especificado y detallado anteriormente en el apartado de Estándares de Código, la aplicación ha sufrido una profunda estandarización de su código fuente así como de los paquetes que componen la misma. En ésta sección se procede a describir con más detalle la distribución de los paquetes y el significado de los mismos:

- **acide:** contiene el código fuente de ACIDE en sí.
  - **acide.configuration:** contiene todo el código relativo al gestor de la configuración detallado más abajo.
  - **acide.factory:** contiene las clases factoría de la aplicación para la creación de componentes. En este caso contiene solamente la *AcideGUIFactory* encargada de la creación de componentes relativos a la GUI de la aplicación.
  - **acide.files:** contiene los componentes necesarios para la gestión de los archivos en la aplicación ya sean de texto, binarios o archivos de proyectos en ACIDE.
  - **acide.gui:** contiene las clases que se corresponden con la interfaz gráfica del usuario de la aplicación. Cada uno de los subpaquetes de la misma se explicarán con más detalle en la sección de interfaz gráfica en el presente documento.

- **acide.language:** contiene las clases necesarias para la gestión de los diferentes lenguajes en ACIDE.
- **acide.log:** contiene las clases necesarias para la gestión del log de la aplicación.
- **acide.main:** contiene la clase de entrada a la aplicación.
- **acide.process:** contiene todos los procesos (hilos) que se ejecutan en la aplicación. Éste a su vez se divide en los siguientes subpaquetes:
  - **acide.process.externalCommand:** contiene los procesos necesarios para mostrar un comando ejecutado en una consola definida por el usuario en una ventana externa.
  - **acide.process.console:** contiene los procesos para el panel de la consola en la aplicación.
  - **acide.process.execution:** contiene el proceso para la ejecución de proyectos en la aplicación.
  - **acide.process.gui:** contiene la ventana de progreso para los procesos de la aplicación.
  - **acide.process.parser:** contiene todas las clases necesarias para el analizador sintático de la aplicación.
- **acide.resources:**
- **acide.utils:** contiene clases con utilidades o pruebas para la ayuda del desarrollo de nuevas funcionalidades.

- **com:** contiene los recursos necesarios para el uso de **XStream**.
- **org:** contiene los recursos necesarios para el **log** de la aplicación.

8.2 MEJORAS GRÁFICAS EN LA APLICACIÓN

Como también se ha comentado con anterioridad, las mejores gráficas en la aplicación han consistido en lo siguiente:

- Adición de iconos a la barra de menús y menús contextuales.
- Adición de icono de ACIDE a las diferentes ventanas de la aplicación para eliminar el icono por defecto de java.

- Adición del logo tipo de ACIDE a la *ventana de Splash* y a la opción de *About Us.*
- Cambiados los iconos para los *archivos normales, compilables* y *principales* tanto en el árbol del explorador de archivos y editor de archivos.
- Añadido el icono de cierre de pestañas en el editor de archivos.
- Cambiados los iconos de la barra de herramientas estática o barra de menús por otros más actuales.
- Unificación en el formato de todas las ventanas de configuración de la aplicación consistente en:
    - Panel principal con el contenido de la ventana.
    - Panel de botones en la parte inferior de la ventana con **FlowLayout** alineando los botones a la derecha.

Para cada apartado de la GUI se detallarán más en profundidad en su sección correspondiente sus mejores gráficas correspondientes.

8.3 Refactorización del Código Fuente

En este apartado se procede a describir con más detalle y en profunidad todas las refectorizaciones que han tenido lugar en el desarrollo del proyecto. Por refactorización se entiende renombrado, rehubicación, adición o eliminación de variables, constantes, métodos, clases, paquetes, ect en el código fuente de una aplicación.

Toda la aplicación ha sufrido una profunda remodelación en cuanto a la distribución de paquetes, métodos, clases y variables principalmente. Aunque toda la aplicación se ha visto afectada por dichos cambios, éstos han sido especialmente notorios en dos componentes específicos de la interfaz gráfica de usuario: el **menú** y el **editor de archivos**.

A continuación se procede explicar con más detalle en qué ha consistido cada uno de ellos:

8.3.1 REFACTORIZACIÓN DEL EDITOR DE ARCHIVOS

Para la gestión del editor de archivos en las versiones previas en la aplicación se tenían dos clases principales en el paquete *gui.editor*: **CreadorEditor** y **Editor**. La primera de ellas se encargaba de la gestión del panel de pestañas que controlan los archivos en el mismo mientras que la segunda de ellas controlaba todo lo relativo a los paneles que muestran el contenido de los archivos en sí.

Ahora el **CreadorEditor** se llama **FileEditorManager** y se encarga de las mismas tareas que se encargaba con anterioridad.

A su vez se ha separado la gestión de la UI del panel de pestañas en otra clase diferente llamada **AcideFileEditorTabbedPaneUI** en el que también se controlan los botones para cerrar las pestañas en el panel de pestañas.

Con respecto a la antigua clase editor se ha subdividido en las clases **AcideFileEditorPanel** que engloba a los dos vistas en las que se divide la zona de edición de texto en los archivos que ahora se llama **AcideFileEditorTextEditionArea.**

El menú contextual también ha sido refactorizado y extraído en otra clase aparte llamada **AcideFileEditorPopupMenu.**

La antigua clase **Editor** controlaba todas las clases anteriores aplicando la técnica de duplicación de código, es decir, como habían dos vistas del editor "simplemente" se copiaba el código asociado a cada uno de ellos dos veces.

La estructura de paquetes que ha resultado así como su descripción se detalla a continuación:

- *acide.gui.fileEditor.fileEditorManager:* contiene la clase **AcideFileEditorManager**.
    - *acide.gui.fileEditor.fileEditorManager.listeners:* contiene las clases que controlan los listeners del gestor del panel de pestañas del editor de archivos.
    - *acide.gui.fileEditor.fileEditorManager.utils:* contiene las clases que controlan las partes necesarias para la gestión del gestor del

panel de pestañas del editor de archivos. Se han separado en dos ramas:

- *acide.gui.fileEditor.fileEditorManager.utils.gui:* Contiene las clases **AcideDragAndDropTabbedPane** y **AcideLineNumberComponent.** La primera de ellas gestiona el panel de pestañas en sí, mientras que la segunda controla el componente que muestra el número de líneas del archivo en el editor de archivos.
- *acide.gui.fileEditor.fileEditorManager.utils.logic:* Contiene la parte lógica del editor de archivos, incluyendo la gestión de la UI y la de los botones de cierre de las pestañas en el panel de pestañas del editor de archivos.

- *acide.gui.fileEditor.fileEditorPanel:* contiene las clases **AcideFileEditorPanel** y **AcideStyledDocument.**
  - *acide.gui.fileEditor.fileEditorPanel.fileEditorTextEditionArea:* contiene la clase **AcideFileEditorTextEditionArea**.
  - *acide.gui.fileEditor.fileEditorPanel.listeners:* contiene las clases que controlan los listeners del panel del editor de archivos**.**
  - *acide.gui.fileEditor.fileEditorPanel.popup:* contiene la clase **AcideFileEditorPopupMenu.**
    - *acide.gui.fileEditor.fileEditorPanel.popup.listeners:* contiene las clases que controlan los listeners para las opciones del menú contextual del panel del editor de archivos.

Tras ésta nueva distribución se tiene un código absolutamente modular, con cada clase separada en su paquete correspondiente por intención y funcionalidad, haciéndolo más **manejable y adaptable** ante futuros cambios en los mismos.

8.3.2 REFACTORIZACIÓN DEL MENÚ

Junto con el editor de archivos, la otra gran refactorización que se ha realizado en la aplicación ha sido la de la clase menú.

En las versiones anteriores toda la gestión del menú se realizaba en la clase **Menu** en el paquete *gui.menus.*

El problema que esto sugiere de cara al desarrollador es simple: ante modificaciones generales o masivas se hacía insostenible mantener todas las funcionalidades del menú. Al principio del desarrollo del proyecto habían ~70 opciones de menú diferentes; por cada una de esas opciones se controlaba además su **ActionListener** correspondiente, por lo que a la hora de identificar dónde estaba cada recurso se hacía un poco caótico.

Por lo tanto se ha decidido separar cada clase correspondiente en su paquete correspondiente siguiendo las siguientes pautas:

1. Cada opción del menú se guardan en una clase diferente y en un paquete dentro de *acide.gui.menuBar.*
2. Cada ActionListener asociado a cada opción del menú está en una clase separada dentro del paquete correspondiente en el subpaquete *listeners.*
3. Si contiene ventanas de configuración u otros recursos gráficos éstos se encuentran dentro del subpaquete *gui.*
4. Si contiene otras clases que implementan otra serie de funcionalidades se incluyen en otro subpaquete *utils.*

A su vez en cada una de esas clases del menú se aplican los estándares de código descritos con anterioridad en el presente documento (ver sección **4.3**).

A continuación se adjunta un ejemplo práctico de cómo ha resultado en la versión final del código:

- *acide.menuBar.configurationMenu.consoleMenu*
    - *acide.menuBar.configurationMenu.consoleMenu.gui*
    - *acide.menuBar.configurationMenu.consoleMenu.gui*

8.4 Gestor de la Configuración

A continuación se detallan los cambios realizados sobre el gestor de la configuración de ACIDE.

8.4.1 Configuración de Proyectos

El gestor de la configuración de proyectos engloba a toda la gestión de los proyectos en ACIDE. Tras las mejoras introducidas en la aplicación, ha sido necesaria una modificación en el mismo.

Durante una gran parte del desarrollo del proyecto, y sobretodo tras la implementación del gestor del espacio de trabajo, algunos recursos que anteriormente estaban controlados bajo la configuración de proyectos, desaparecieron en releases intermedias de la aplicación.

En un principio se decidió respetar la gestión de proyectos anterior, incluyendo en la misma:

- **Nombre de proyecto.**
- **Ruta del archivo de configuración.**
- **Ruta y directorio del ejecutable del panel de la consola.**
- **Configuración de la ventana que incluyen:**
  - **Tamaño.**
  - **Localización.**
  - **Localización del separador del panel horizontal y vertical en la ventana.**
- **Lista de archivos relativos al proyecto.**

Finalmente y a los recursos anteriormente mencionados se han añadido los siguientes recursos:

- *Parámetros de configuración de presentación del panel de la consola*.
  - **Tipo de fuente.**
  - **Estilo de fuente.**
  - **Tamaño de fuente.**

    - **Color de fuente.**
    - **Color de fondo.**
    - **Tamaño del buffer en número de líneas.**
- *Parámetros de configuración del compilador asociado al proyecto.*
    - **Ruta del compilador.**
    - **Argumentos del compilador.**
    - **Flag de compilación de todos los archivos del proyecto.**
    - **Separador de archivos.**
    - **Extensión de los archivos a compilar.**

Con ésto último además se consigue que se carguen dichos parámetros en la ventana de configuración del compilador y que el usuario se ahorre el tener que definir los mismos de nuevo.

También se han introducido dos variables que almacenan las rutas de los últimos archivos y proyectos abiertos por el usuario, facilitando de ésta forma la selección de archivos y proyectos al usuario de una forma más rápida.

Siguiendo el hilo de facilitar las cosas al usuario, también se ha optado por añadir sendas opciones en el menú que aceleran el proceso de apertura y adición de archivos a un proyecto y son respectivamente:

- **Abrir todos los archivos:** abre todos los archivos relativos a un proyecto en el editor de archivos. Si ya está abierto simplemente se ignora.
- **Añadir todos los archivos abiertos:** añade todos los archivos abiertos en el editor de archivos al proyecto. Si ya está añadido simplemente se ignora.

También se ha habilitado la *multiselección* de archivos a la hora de añadir los archivos a un proyecto o de abrirlos en ACIDE para agrupar el proceso múltiple en uno sólo.

El funcionamiento de la configuración por defecto se ha visto ligeramente modificada. Se ha tenido en cuenta que un proyecto por defecto no tiene archivos asociados y que éstos deben ser gestionados por el gestor del espacio de trabajo,

por tanto en la configuración por defecto siempre la lista de archivos asociados estará vacía.

8.4.2 CONFIGURACIÓN DE BARRA DE MENÚS

La configuración de la barra de menús así como gestión ha sido profundamente rediseñada. Anteriormente, dicha configuración guardaba una lista estática con cada una de las opciones de menú correspondientes y su archivo de configuración presentaba el siguiente aspecto:

```
true
true
...
true
false
```

Esto planteaba ciertos riesgos en cuanto al diseño, porque provocaba que el desarrollador tocase demasiados aspectos a la hora de añadir una nueva opción al menú. También hacía todo muy caótico ya que cada opción estaba por separado y no había ninguna correlación entre las opciones del menú similares; por ejemplo, *"guardar archivo como"* podía estar en la posición 52 del array mientras que *"guardar archivo"* en la posición de 2.

Se decidió pues, adoptar en primer lugar la medida de hacer que el gestor de la configuración de los menús tuviera una lista dinámica ó **ArrayList** que fuera la que guardase todas las opciones del menú. Ésta circunstancia provocó que se tuviera que definir para cada opción del menú un nombre único que lo identificara en orden de acceder al valor del mismo en la lista y que fuera independiente del lenguaje actual en la aplicación. Por tanto, se decidió definir dichos nombres en Inglés, provocando que ahora el archivo de configuración de menús tenga el siguiente formato:

```
New File = false
Open File = false
Open Recent Files = false
Open All Files = false
Save File As = false
```

```
Save File = false
...
Show Log Tab = false
Show Explorer Panel = false
Show Console Panel = false
Show Help = false
Show About Us = false
```

Pero, ¿Qué ventajas tiene todo esto? La principal ventaja reside en la facilidad con la que el desarrollador tiene ahora para agregar opciones al menú de una forma clara y sencilla. Bastará con programar la nueva entrada siguiendo los esquemas ya mostrados en el código sin necesidad de editar los archivos de configuración. Los pasos para añadir una nueva opción al menú ahora son:

1. Definir sus constantes de nombre e icono (si tiene):

```
public static final String NOMBRE_NAME = "New Name";
public static final ImageIcon NOMBRE_IMAGE = new ImageIcon("icon
path");
```

2. Crear la opción correspondiente (JMenuItem, JCheckBoxMenuItem ó JMenu) en el menú correspondiente en el método *buildComponents().*
3. Añadir la nueva opción al menú mediante el método *addComponents().*
4. Establecer su etiqueta en los lenguajes disponibles y establecerla junto con su atajo de teclado en el método *setTextOfMenuComponents().*
5. Configurar su visibilidad en el método *updateComponentsVisibility().*
6. Establecer su **Action Listener** en el método *setListeners().*

No obstante, para que dicha opción esté visible se tendrá que editar la entrada correspondiente en la ventana de configuración de los menús, cuya profunda reestructuración se abordará más adelante en el presente documento así como el rediseño de la visibilidad de los menús y los cambios y medidas que ahora se han aplicado durante el desarrollo del proyecto.

8.4.3 CONFIGURACIÓN DE BARRA DE HERRAMIENTAS

La configuración de la barra de herramientas ha sufrido importantes modificaciones en esta versión de la aplicación. Anteriormente, la configuración de la barra de tareas se basaba sólo y exclusivamente en la barra de comandos ejecutados en el panel de la consola.

Actualmente, se ha dividido la misma en 3 partes diferentes:

- **Barra de herramientas basada en la barra de menús:** ejecuta opciones relativas a la barra de menús de ACIDE. Por falta de tiempo, no ha sido posible realizar una ventana de configuración para modificarla.
- **Barra de herramientas de comandos para el panel de la consola:** ejecuta comandos en el panel de la consola. No obstante también se han añadido
- **Barra de herramientas para ejecución de aplicaciones externas:** ejecuta aplicaciones externas a ACIDE, pensada para tener acceso a programas relacionados con los proyectos con los que se esté trabajando.

A su vez dentro de los comandos ejecutados en el panel de la consola de ACIDE se han introducido la funcionalidad de solicitar al usuario un parámetro extra para la ejecución de losi mismos que puede ser:

- **Ninguno:** el comando se ejecuta sin parámetro extra.
- **Texto:** la aplicación muestra un cuadro de texto al usuario para añadirlo al comando en sí definido en el campo *action*.
- **Archivo:** la aplicación muestra una ventana de selección de archivos al usuario para añadirlo al comando en sí definido en el campo *action*.
- **Directorio:** la aplicación muestra una ventana de selección de directorios al usuario para añadirlo al comando en sí definido en el campo *action*.

Ésto ha provocado obviamente el cambio en el formato del archivo de configuración de la barra de herramientas. Ahora mismo, dicho archivo contiene por cada comando:

```
// Comentario
name = ...
action = ...
hintText = ...
icon = ...
parameterText = NONE|FILE|DIRECTORY|TEXT
isExecutedInSystemShell = true|false

...

// End of Console Tool Bar Button Configuration
...
// End of External Applications Tool Bar Button Configuration
```

Nótese que para el parseado de dicho archivo se ha seguido la política anterior de gestión de excepciones cuando el mismo contiene un formato incorrecto.

8.4.4 CONFIGURACIÓN LÉXICA

La configuración léxica ahora se aplican a cada archivo en el editor de archivos y no a los proyectos en general. Para hacer esto posible, cada panel de editor de archivos tiene su configuración léxica asociada y es la misma que se usa en su AcideDocument que se encarga de aplicar la configuración léxica a los archivos abiertos en el editor.

La configuración léxica sigue compuesta por los siguientes módulos:

- *Gestor de limitadores*.
- *Gestor de comentarios* → En ésta nueva versión se han introducido además la posibilidad de la configuración del formato de los mismos.
- *Gestor de tokens y palabras reservadas*.
- *Gestor de extensiones de archivo*.

Para la gestión de los archivos de configuración XML se ha seguido usando **XStream** para el parseado de los mismos en clases JAVA.

Para un correcto tratamiento de dichos ficheros, ha sido necesario configurar el **XStream** A veces dicho parseo daba fallos en cuanto al driver que utilizaba para realizar sus operaciones y para solucionarlo se ha optado por la aplicacion del driver **DomDriver.**

Uno de los aspectos duros de corregir ha sido el marcado de paréntesis, llaves y corchetes en el editor de archivos. En general también se han corregido los problemas con la aplicación del estilo de los comentarios: con anterioridad, tanto los tokens como el marcado de paréntesis, llaves y corchetes no se realizaba correctamente si se encontraban en los comentarios. Ahora se mantiene su estilo actual, es decir si es palabra reservada se encuentra en un comentario recuperará su estilo cuando el comentario sea eliminado. Del mismo modo, cuando desaparece el marcado de paréntesis, llaves y corchetes el estilo de comentario prevalece.

Por último, para la definición de grupos de tokens o palabras reservadas también se ha cambiado el formato de sus nombres para estandarizarlos en los archivos de configuración XML de la siguiente forma:

```
Color: [R: 0, G: 0, B: 0], Font Style: Bold|Italic|Italic and Bold,
Case Sensitive: Yes|No
```

8.4.5 Configuración Sintáctica

La configuración sintáctica fue en principio uno de los grandes objetivos del proyecto. No obstante por falta de tiempo y ante la preferencia que se dió a otras tareas, éste apartado no ha sido desarrollado en profundidad durante en el transcurso del desarrollo del proyecto.

La configuración sintáctica ahora se aplica a cada archivo en particular en el editor de archivos. Una de las mejoras introducidas relacionadas a la configuración sintáctica ha sido la corrección del proceso de generación de la gramática en la ventana de configuración de la misma.

Con anterioridad, la aplicación se quedaba colgada al realizar el proceso en el mismo hilo de ejecución que la ventana de la aplicación. Lo que se ha propuesto para solventarlo ha sido el desarrollo de un nuevo proceso en el que se muestra el proceso en una ventana de progreso y que impide la congelación de la aplicación.

Dicho proceso ha sido separado en los siguientes pasos:

1. Se copian los archivos de categorías léxicas y las reglas sintáctica al directorio raíz de la aplicación.
2. Ejecución de **antlr** para la obtención de los archivos .java a partir del archivo de la gramática.
3. Modificación del archivo **GrammarParser.java.**
4. Compilación de los archivos generados.
5. Copiado de los archivos correspondientes a la carpeta *acide/process/parser/grammar/*
6. Generación del archivo JAR.
7. Borrado de archivos intermedios del proceso.
8. Copiado de archivo JAR al directorio *configuration/grammar/*

8.4.6 CONFIGURACIÓN DEL ESPACIO DE TRABAJO

Éste componente ha sido uno de los puntos fuertes de la nueva versión de la aplicación. Dicho componente se encarga de la gestión del espacio de trabajo de ACIDE. Por espacio de trabajo se entiende todos los parámetros de configuración que no pertenecen a la configuración de los proyectos, englobando los siguientes componentes:

8.4.6.1 CONFIGURACIÓN DEL EDITOR DE ARCHIVOS

Inicialmente y ante la necesidad de guardar recursos relacionados con el editor de archivos de la aplicación, surgió la idea de implementar un gestor que se encargara de guardar toda la configuración relativa al editor de archivos ya que dichos parámetros no pertenecían en un principio a la configuración de los proyectos.

De este modo, éste gestor llamado **AcideFileEditorConfiguration** guarda los siguientes parámetros:

- **Lista de archivos abiertos** en el editor de archivos desde la última vez que se cerró ACIDE. A su vez cada uno de éstos archivos abiertos guarda la siguiente configuración:
  - **Ruta del archivo**.
  - **Tipo de archivo**: Normal, Compilable o Principal.
  - **Posición del cursor.**
  - **Posición del divisor horizontal de las dos vistas de archivo.**
  - **Vista activa del editor.**
  - **Configuración léxica:** Puede tener una configuración léxica distinta a la definida por defecto, por eso se hace indispensable su guardado.
  - **Configuración sintáctica previa.**
  - **Configuración sintáctica actual.**
- **Nombre del editor activo:** En un principio se guardaba el índice del mismo, pero debido a que en el editor de archivos no siempre se

guardan o se cargan todos los archivos abiertos al cerrar la aplicación, se optó por este diseño, sacrificando el orden de los mismos en el editor.

- **Nombre de fuente.**
- **Estilo de fuente.**
- **Tamaño de fuente.**
- **Color de fuente.**
- **Color de fondo.**
- **Tipo de modo de escritura:** Sobreescritura o inserción.
- **Sangrado automático:** Indica si se aplica el sangrado automático o no.
- **Número máximo de líneas para enviar a la consola:** Éste parámetro sirve como medida de seguridad para el usuario. Si el número de líneas supera éste número, se le solicita una confirmación al mismo cuando éste selecciona la opción de enviar el contenido del archivo activo al panel de la consola. De éste modo se evita, por ejemplo, el envío automático de archivos grandes cuando el usuario selecciona la opción de envío por equivocación.
- **Line wrapping:** Indica si se aplica line wrapping.

Para la carga de la configuración del editor de archivos se ha diseñado la clase **AcideFileEditorLoader,** que se encarga de cargar la configuración del editor de archivos de la siguiente forma en los siguientes pasos:

- Carga los archivos relativos a la *configuración del proyecto*, en el orden en el que éstos estén en la lista.
- Después revisa cada uno de los archivos de la lista de archivos abiertos en la *configuración del espacio de trabajo del editor de archivos*:
  - Si está abierto entonces se **actualiza** su configuración con la guardada en el espacio de trabajo para el editor de archivos.
  - Si no está abierto entonces se **abre** con la configuración guardada en el espacio de trabajo para el editor de archivos.
- Por último se selecciona el archivo activo con el nombre guardado en el configuración del espacio de trabajo del editor de archivos.

Un detalle importante en todo éste proceso es la **comprobación de la existencia** de los archivos guardados tanto en la *configuración de los proyectos* como en el *editor de archivos*. Si dichos archivos no existen porque han sido borrados o bien han sido movidos a otra ubicación, se eliminarán de dichas configuraciones y se mostrarán sendos mensajes de advertencia.

8.4.6.2 Configuración de Archivos Recientes

Guarda la configuración relativa a la **lista de archivos recientes** de la aplicación mediante un ArrayList de rutas de archivos.

8.4.6.3 Configuración de Proyectos Recientes

Guarda la configuración relativa a la **lista de proyectos recientes** de la aplicación mediante un ArrayList de rutas de archivos.

8.4.6.4 Configuración de Léxicos por Defecto

Una de las novedades importantes de la nueva versión de la aplicación es la posibilidad de la definición de una serie de configuraciones léxicas por defecto para los archivos en función de la extensión de los mismos. El módulo encargado de realizar esa tarea se llama **AcideLexiconAssignerConfiguration** y a continuación se procede a detallar los parámetros que gestiona:

- **Lista de configuraciones léxicas por defecto:** Guardan todas las configuraciones léxicas por defecto de la aplicación. Cada configuración léxica por defecto a su vez guarda los siguientes parámetros:
  - **Descripción.**
  - **Lista de extensiones asociadas.**
  - **Configuración léxica asociada.**
- **Configuración léxica para el panel de la consola.**
- **Aplicación de la configuración léxica para el panel de la consola:** Determina si la configuración léxica por defecto definida para el panel de la consola se debe aplicar al mismo o no.

Durante su desarrollo también se hizo especial hincapié en la **validación** de las configuraciones léxicas introducidas por el usuario. En caso que una configuración léxica seleccionada no exista o sea incorrecta, el gestor actuará previniendo posibles errores y cargará la configuración léxica por defecto.

8.4.6.5 CONFIGURACIÓN DE PANEL DE LA CONSOLA

Para mantener la concoordancia entre el panel de la consola y el editor de archivos se ha optado finalmente por definir la configuración del panel de la consola para la configuración del espacio de trabajo y no guardar sus parámetros a nivel de proyecto.

Dicha configuración contiene solamente el siguiente parámetro:

- **Configuración léxica:** Puede tener una configuración léxica distinta a la definida por defecto, por eso se hace indispensable su guardado.

8.4.6.6 GESTIÓN DE ERRORES

Para la gestión de errores y para la prevención de comportamientos indeseados en la aplicación, se definió como convenio que cualquier error encontrado en el archivo XML de configuración del espacio de trabajo, provoca que se cargue la **configuración del espacio de trabajo vacía**, es decir, se pierden las configuraciones afectadas por el espacio de trabajo mencionadas con anterioridad, pero se asegura que la aplicación no quede colapsada en su ejecución.

8.5 INTERFAZ GRÁFICA

Tanto a nivel gráfico como a nivel lógico, la interfaz gráfica de la aplicación ha sufrido números cambios en cada uno de sus componentes durante el desarrollo del proyecto. A continuación se procede a detallar en profundidad dichos cambios para cada uno de los componentes de la misma:

8.5.1 VENTANA PRINCIPAL

Al igual que para todas las ventanas de configuración de la aplicación, la ventana ha sido estandarizada con los métodos anteriormente explicados en el capítulo de *estándares de código fuente* (ver **apartado 4.3** del presente documento). La ventana principal ahora se encuentra lanzada dentro del hilo de ejecución de Swing llamado **EventDispatchThread**, evitando comportamientos indeseados al inicio de la misma como por ejemplo, problemas con los focos o los cursores de texto. A continuación se detallan los cambios más significativos aplicados sobre cada uno de los componentes de la ventana principal de la aplicación:

8.5.1.1 BARRA DE MENÚS

La barra de menús ha sido con mucha diferencia y junto al editor de archivos del aplicación que más modificaciones ha sufrido con esta nueva versión de ACIDE.

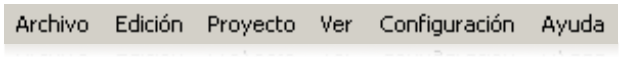
Todas las modificaciones relativas a la configuración, estandarización y refactorización han sido ya expuestas con anterioridad por lo que en ésta sección se describirán otras tareas realizadas sobre el mismo.

Como principal novedad introducida en ésta nueva versión se encuentra el modo en el que la configuración de los menús es aplicada sobre la barra de menús.

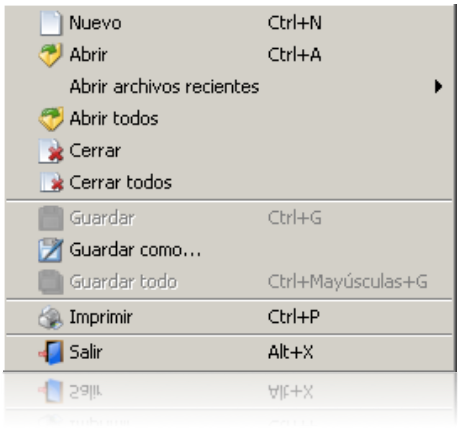
Con anterioridad, la barra de menús era generada desde cero una y otra vez. Ahora la visibilidad de las opciones de la barra de menús es gestionada mediante los métodos *setVisible()* de cada componente de la clase menú. La principal ventaja de ésto radica en que ahora en todo momento la ventana de configuración de la configuración de los menús puede ser generada automáticamente en función de los

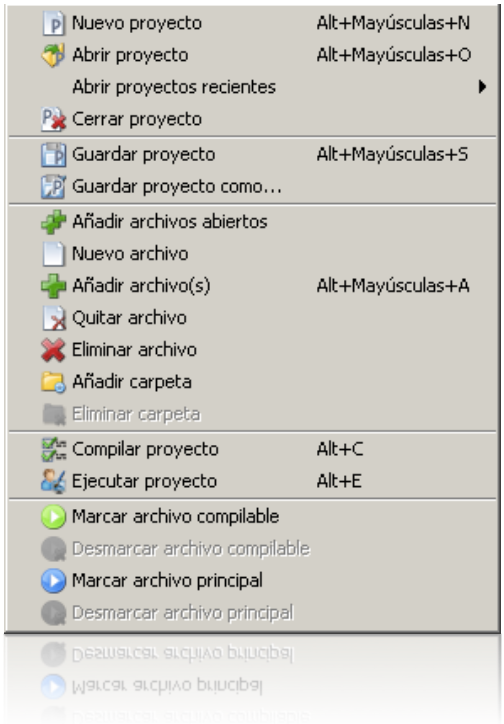
menús que se encuentran añadidos a la barra de menús. Como se ha mencionado anteriormente, la forma previa de aplicar los cambios de la configuración sobre la barra de menús provocaban que a efectos prácticos, la opción no existiese en el menú por lo que impedía que la ventana de configuración se generase automáticamente con dichas opciones que no estaban añadidas a la barra de menús.

A continuación se muestra el aspecto que tiene la barra de menús de la aplicación, que mantiene el formato de la anterior:



Y alguno de los submenús de la aplicación:

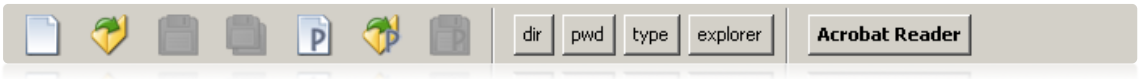




8.5.1.2 BARRA DE HERRAMIENTAS

La barra de herramientas de la aplicación ha sufrido importantes cambios; muchos de los cuales ya han sido descritos con anterioridad como por ejemplo, la el cambio de los iconos en la barra de herramientas basada en los menús, la división en diferentes secciones y así como la adición de parámetros extra para la barra de herramientas de comandos ejecutados en el panel de la consola de la aplicación.

Para la barra de aplicaciones externas, se ha optado finalmente por mostrar la el nombre del comando en negrita y así hacer una distinción gráfica con respecto a la barra de herramientas basada en comandos que se ejecutan en el panel de la consola.



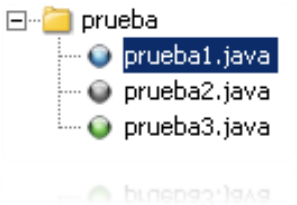
Otra de las novedades significativas ha sido la adición de dos botones de scroll a la misma para garantizar el acceso a todos los contenidos de la misma cuando éstos son más grandes que el tamaño de la ventana principal como se puede apreciar a continuación:

8.5.1.1 PANEL DEL EXPLORADOR DE ARCHIVOS

El explorador de archivos ha sufrido mejoras en el aspecto gráfico, en cuanto a la sustitución de los antiguos iconos por unos nuevos más modernos, respetando el color gris para los archivos normales, verde para los archivos compilables y el azul para los principales.

La principal novedad que presenta es la comprobación de la existencia de carpetas con el mismo nombre en el el mismo directorio donde se pretende crear otra con el mismo nombre. Anteriormente ésta característica no estaba disponible y podría llevar al usuario a ciertos equívocos por lo que ahora se lanza el siguiente mensaje de aviso:

Finalmente el aspecto del panel del explorador de archivos con un proyecto cualquiera podría ser:



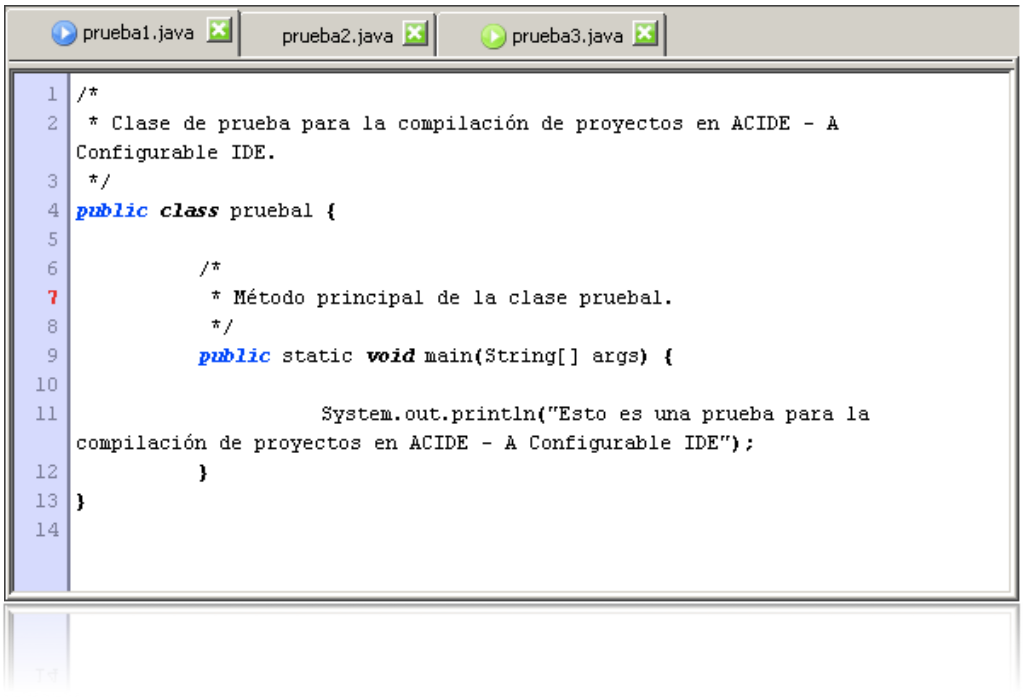
8.5.1.2 EDITOR DE ARCHIVOS

El editor de archivos es, junto con la barra de menús la parte de la aplicación que más cambios ha sufrido durante el transcurso del desarrollo del proyecto.

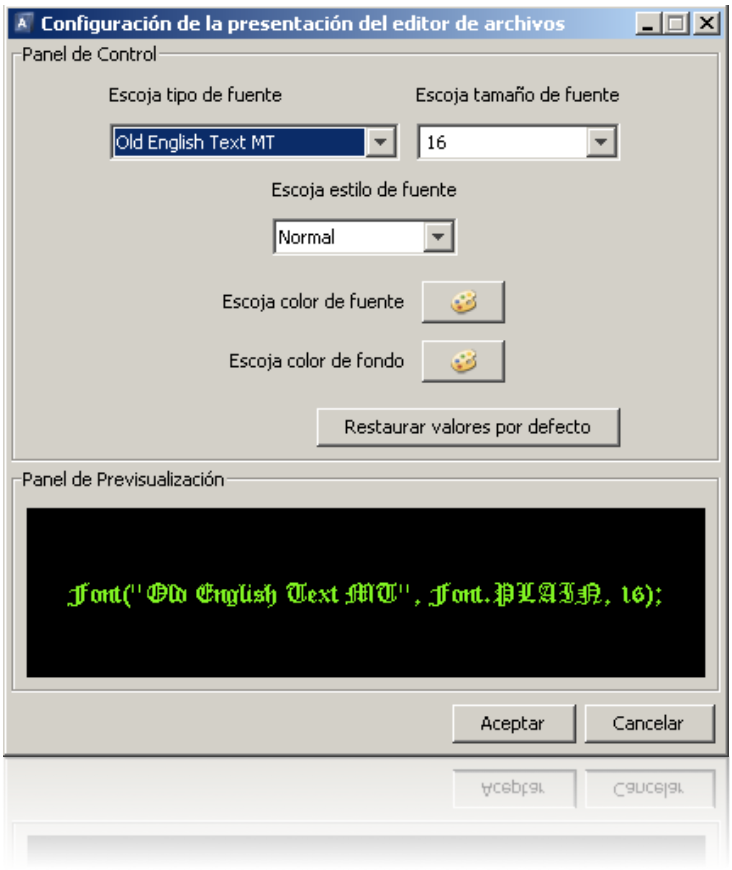
A juicio del autor, ha resultado ser **la parte más delicada** de la aplicación y la que todavía presenta las mayores deficiencias. Prácticamente todos los módulos de la aplicación dependen de él en mayor o menor medida.

Como cambios significativos y que ya se han mencionado con anterioridad, destacan el **cambio de los iconos** para el panel de pestañas, multitud de nuevas funcionalidades y como parte a destacar, la inclusión de un componente que se encarga de *mostrar los números de líneas* de los archivos abiertos en el editor de archivos en la parte izquierda.

El anterior componente que se encargaba de dicha tarea no gestionaba correctamente el *line wrapping* por lo que se optó finalmente por incorporar otro componente que pudiera detectar cuando una línea sigue siendo la misma y por tanto no aumentar el número de línea, además de mostrar solamente las líneas reales del archivo abierto en el editor de archivos y marcar la línea con un color diferente, en este caso el rojo como se muestra a continuación:



Otra novedad importante a destacar es la personalización de la **presentación del editor de archivos** mediante una ventana de configuración, en la cual el usuario podrá modificar el estilo de la fuente así como los colores de la misma, con la posibilidad de restaurar los valores por defecto mediante un botón tal y como se muestra a continuación:



Al hilo de la anterior novedad, el usuario también podrá modificar *el tamaño de la fuente* del editor de archivos mediante la opción **CTRL+Rueda del ratón** con el efecto **zoom**, así como realizar el scroll línea a línea mediante **CTRL+Flecha**.

Junto con éstas funcionalidades, encontramos a continuación un grupo de nuevas opciones añadidas a la funcionalidad del editor de archivos:

- **Ajuste de línea ó line wrapping:** Cuando ésta opción está habilitada, el usuario no podrá escribir más allá de la longitud del campo de texto y la línea actual se separará en dos en vez de mostrar la barra de scroll para seguir editando más allá del límite.
- **Sangrado automático:** Cuando el usuario presiona la tecla ENTER para seguir escribiendo en una nueva línea, seguirá escribiendo en la misma columna que la línea anterior. Lo que falta definir es si dicho sangrado es en tabulaciones o en espacios.
- **Asignar léxico:** Asigna una configuración léxica al archivo activo en el editor de archivos. Ahora como cada configuración léxica pertenece a cada archivo y no a nivel de proyectos se ha decidido añadir dicha opción para los archivos en el editor de archivos.
- **Enviar Contenido al Consola:** Se envía el contenido del archivo activo en el editor de archivos al panel de la consola para su ejecución. Actualmente cada línea se guarda como un *comando separado* en el historial de comandos del panel de la consola, pero en futuras versiones dicha versión podrá ser configurada por el usuario.
- **Definición de máximo número de líneas a enviar a la consola:** Ésta opción se utiliza para preguntar al usuario si está seguro acerca del envío del contenido del archivo activo en el editor de archivos al panel de la consola cuando el número de líneas del mismo supera el número definido por el usuario mediante la presente opción.

8.5.1.3 PANEL DE LA CONSOLA

Desde el principio del desarrollo del proyecto, la mejora del panel de la consola de la aplicación ha sido una tarea **compleja y para nada trivial**.

Para ésta nueva versión, se ha hecho especial hincapié en corregir los problemas de las líneas en blanco extra que se añadían con repetidas pulsaciones de la tecla intro. La manera de mostrar el contenido recibido por la consola y su adición al texto actual también fue otro de los problemas encontrados para adaptar y corregir el funcionamiento de la opción de **echo del comando**. Al hilo de lo anterior también se observó que el texto escrito tras la marca del *prompt* no se sobreescribía tras la pulsación de un comando de la barra de herramientas correspondiente. Otra de las características que chocó al autor fue la ausencia de una opción para **borrar el contenido del buffer** de la misma, así como el de **reset** del ejecutable cargado en el panel de la consola muy útil cuando el mismo se queda colgado en alguna operación interna. Dado que en un principio la propuesta inicial era la de poder enviar la señal de **SIGINT** al ejecutable y ante la imposibilidad de llevarla a cabo por el momento, se pensó en ésta última opción para evitar cuelgues en el ejecutable y poder seguir trabajando con el mismo con normalidad.

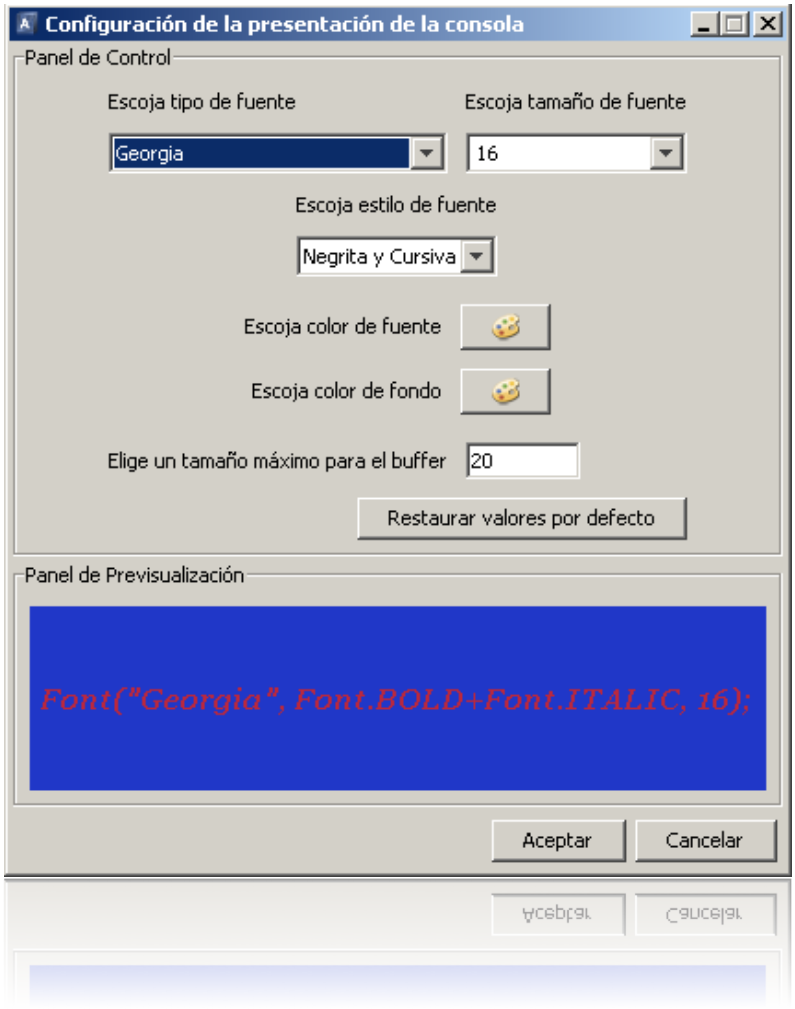
Otro comportamiento que no gustó mucho al autor en un principio fue la *necesidad* de tener abierta una configuración para la consola en todo momento, por lo que se decidió prescindir de ésta característica y más tarde se añadiría la opción de **cerrar** la consola cargada en el panel de la misma. También a la hora de definir la configuración de la consola a cargar, se comprueba que dicha consola existe, evitando así excepciones y comportamientos indeseados.

Siguiendo el análisis de errores solventados y al hilo de lo anterior, se ha corregido también el problema de los procesos relativos a los ejecutables cargados en el panel de la consola. Con anterioridad dichos **procesos** seguían activos en el sistema *consumiendo recursos* del mismo; gracias a las nuevas funcionalidades introducidas, cuando la consola se cierra, ya sea bien por la opción explícita del usuario, por reinicio de la misma o porque la aplicación se cierra, se busca dicho

proceso en el sistema y **se termina**, *liberando los recursos* que el mismo estaba consumiendo.

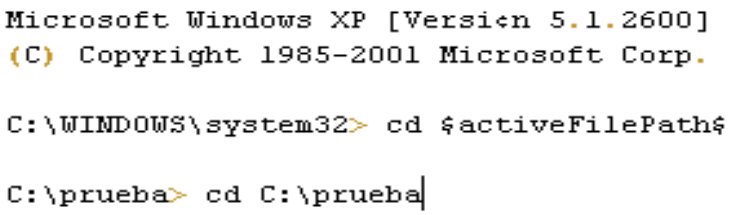
Finalmente, durante la última fase del desarrollo se añadieron dos funcionalidades importantes: **tamaño máximo del buffer en número de líneas** y **limitación de la edición** sobre el panel de la consola solamente detrás de la marca del prompt. *La primera opción* permite mostrar por consola el número de líneas especificado en el tamaño del buffer, consiguiendo que la sobrecarga del sistema no sea muy grande, siempre y cuando el tamaño del buffer no sea relativamente grande. Con anterioridad, a medida que el usuario usaba la consola, el buffer se iba llenando y con ello, provocaba que la aplicación en general se comportara de una forma más lenta. *La segunda opción* anteriormente mencionada, permite que el usuario no pueda borrar el contenido previamente cargado en la misma, al estilo de la consola de MS-DOS. Con anterioridad, si el usuario seleccionaba algún texto en la misma podía borrar el mismo, con los posibles errores que ello podía provocar. Dichos problemas venían acarreados del **borrado de la marca del prompt** de la misma que provocaba que la consola se colapsase invalidando eventos de teclado, ratón y de envío de comandos desde la barra de herramientas.

Al igual que el editor de archivos de la aplicación, el panel de la consola también dispone de una **ventana de configuración de la presentación** del mismo:



Al igual que para el panel de editor de archivos de la aplicación, ahora es posible aplicar una **configuración léxica** al mismo, ya sea por defecto (como ya se ha explicado con anterioridad en el presente documento) ó bien mediante la opción de menú correspondiente. Al hilo de las similitudes anteriormente mencionadas, el panel de la consola también presenta los atajos de *CTRL+Flecha* para el **scroll línea a línea**, así como el efecto **zoom** mediante *CTRL+Rueda del ratón.*

Finalmente el aspecto que presenta el panel de la consola con una configuración normal sería el siguiente:



8.5.1.4 BARRA DE ESTADO

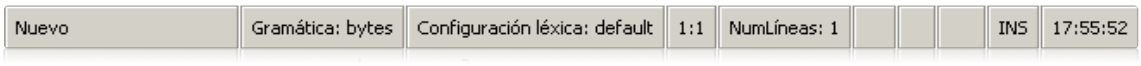
La barra de estado ha sufrido una profunda reestructuración gráfica así como la adición de nuevos cuadros de estado para las distintas opciones que se han ido introduciendo en la aplicación.

El primer cambio ha consistido en el rediseño basado en **JPanel** y no en **JBox** y **JTextFields** como en versiones anteriores. La razón fundamental para el cambio ha sido principalmente el mejor manejo tanto visual como a nivel de tamaño que ofrecen los **JPanels**.

Una vez cambiado su aspecto se han añadido los siguientes paneles de información extra en la misma:

- **Panel de modo de escritura:** "INS"|"OVR" para el modo inserción y sobreescritura respectivamente.
- **Panel de número de líneas:** Muestra el número de líneas del archivo activo en el editor de archivos.

Finalmente, el aspecto que presenta ahora la barra de estados de la aplicación en un estado normal sería el siguiente:



8.5.1.5 SPLASH SCREEN

La pantalla de splash ha pasado por diferentes fases durante el transcurso del desarrollo del proyecto. Desde el principio el objectivo fundamental era el de mostrar una imagen con el logotipo de la aplicación en ella y una barra de progreso que a su vez informase de los recursos que se fueran cargando.

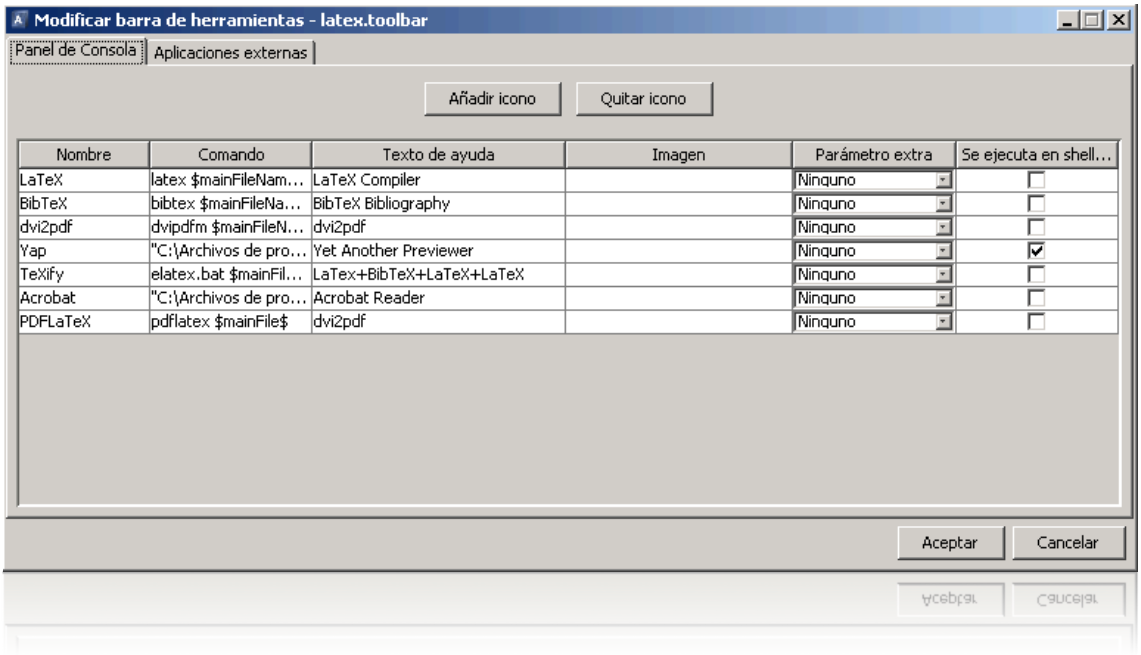
Debido a problemas con hilos de ejecución, principalmente con el **EventDispatchThread,** y ante la posibilidad de que la carga del programa se viera ralentizada, finalmente se ha optado por mostrar solamente la imagen con el logo y además la aplicación se inicia en el hilo de ejecución correcto mediante el método *SwingUtilities.invokeAndWait().*

8.5.1 VENTANAS DE CONFIGURACIÓN

En esta sección se procede a describir los cambios más significativos en las ventanas de configuración de la aplicación:

8.5.1.1 VENTANA DE CONFIGURACIÓN DE LA BARRA DE HERRAMIENTAS

El aspecto que presenta la misma con la nueva versión es el siguiente:



Como principales novedades, a simple vista se puede observar que la ventana contiene un panel de pestañas para cada una de los tipos de barra de herramientas que contiene la aplicación, a falta de añadir la de la barra de herramientas basada en menús como ya se ha mencionado con anterioridad.

Se ha habilitado la edición sobre las mismas tablas, y si se se han producido cambios sobre las mismas se pedirá confirmación al usuario.

Para habilitar la edición sobre las tablas ha sido necesario la definición de clases **TableModel** que hacen que ésto sea posible. También se han implementado **CellEditor** y **CellRenderer** para hacer posible que el usuario pueda editar un **JComboBox** dentro de una celda de la tabla. Para seleccionar la imagen del icono

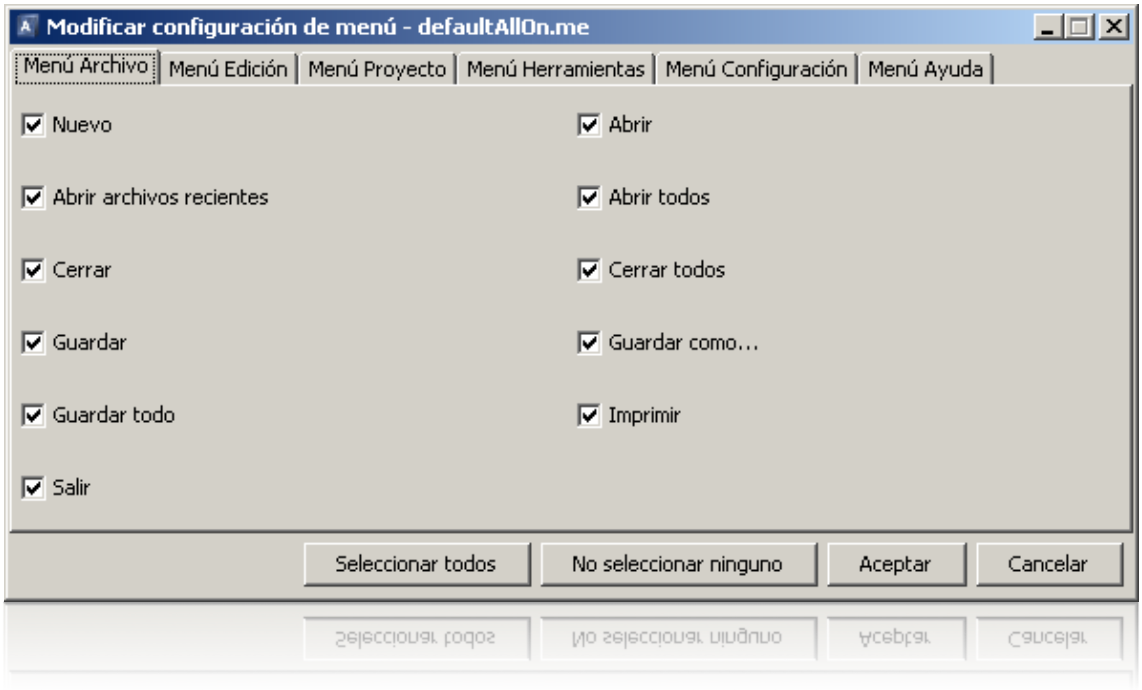
del botón el usuario podrá seleccionar la opción de abrir un archivo mediante la opción del menú contextual disponible en la celda correspondiente.

Otras de las funcionalidades implementadas ha sido la de la selección automática del siguiente elemento de la lista cuando un comando es eliminado de la misma, permitiendo al usuario agilizar las operaciones sobre las misma.

8.5.1.1 VENTANA DE CONFIGURACIÓN DEL MENÚ

Ésta ventana de configuración ha sido rediseñada por completo, tanto en aspecto gráfico como en el modo en que se genera la misma.

Con la nueva versión de la aplicación, la ventana de configuración del menú tiene el siguiente aspecto:

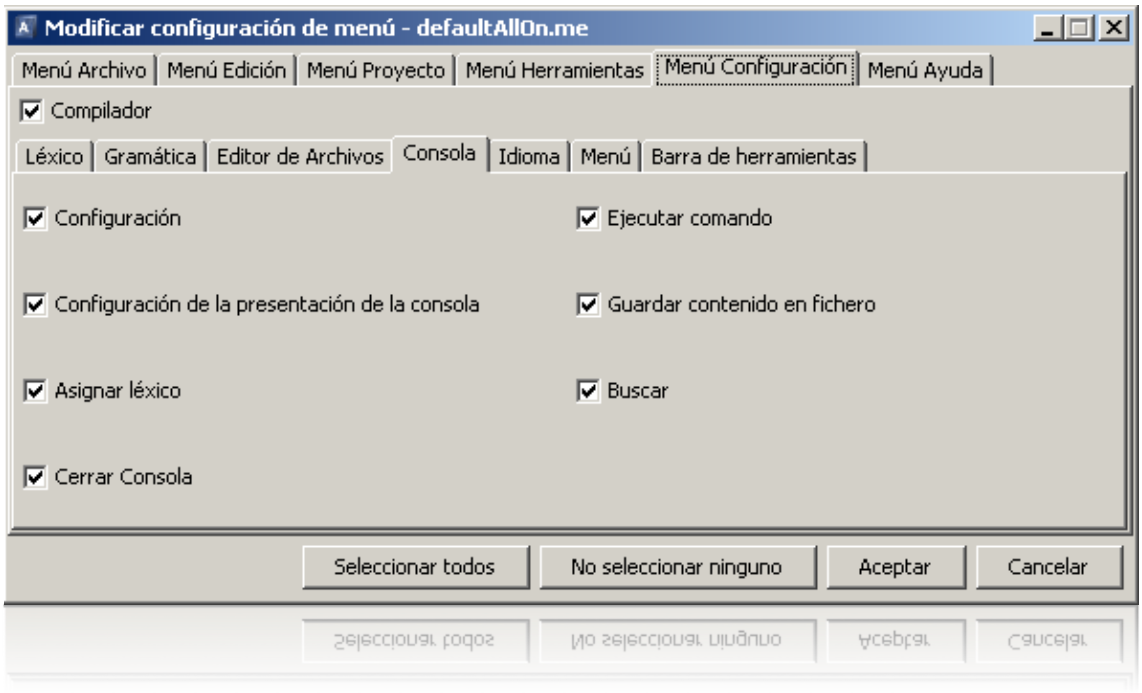


Como se puede apreciar a simple vista, la ventana ha sido dividida en diferentes pestañas, cada una correspondiente a una opción de la barra de menús en su nivel más superior.

En el interior de cada panel de pestañas se genera un panel que contiene **JCheckBox** como la anterior ventana, con la salvedad de que dichos check box son

objetos generados automáticamente en función del contenido del menú correspondiente.

Un caso especial es el del **menú configuración**:



En este caso en particular y dado que todas las opciones del menú son submenús excepto la opción del compilador, se ha hecho un poco de trampa en este sentido para adecuar la ventana a las necesidades del menú. Por otra parte cada uno de los submenús a su vez está generado de una forma automática.

8.5.1.2 VENTANA DE CONFIGURACIÓN DE LA CONFIGURACIÓN LÉXICA

La ventana de configuración léxica presenta pocos pero significativos cambios con respecto a las anteriores versiones de la aplicación.

El aspecto actual de la misma es el siguiente:

A simple vista, y siguiendo las mismas pautas que en las anteriores ventanas de configuración, se ha optado por dividir la misma en diferentes pestañas, cada una

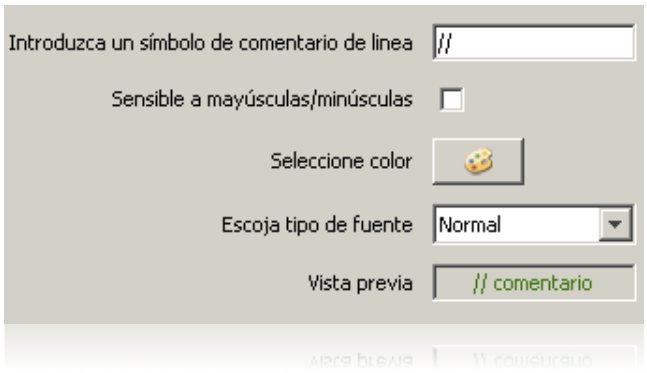
de ellas con sus propias funcionalidades e independientes de las otras. La anterior presentación de la misma producía a primera vista una sensación de saturación al usuario, ya que tenía a su alcance múltiples opciones presentadas a la vez con el consecuente impacto visual que esto producía.

Con respecto al funcionamiento de la misma, se ha decido situar el botón de **guardar como** en la misma ventana de configuración de la misma y eliminarla de la barra de menús. Cuando el usuario selecciona el botón **aplicar** los cambios serán automáticamente guardados en el archivo de configuración correspondiente, lo que ha propiciado que la opción de **guardar** haya desaparecido también del menú de *configuración léxica*.

Cuando el usuario pulsa *ESC* si ha habido cambios en cualquiera de los tres paneles, pregunta al usuario si quiere guardar los cambios.

Como cambio significativo en cuanto a la adición de nuevas funcionalidades en los paneles destacan dos cambios por encima del resto:

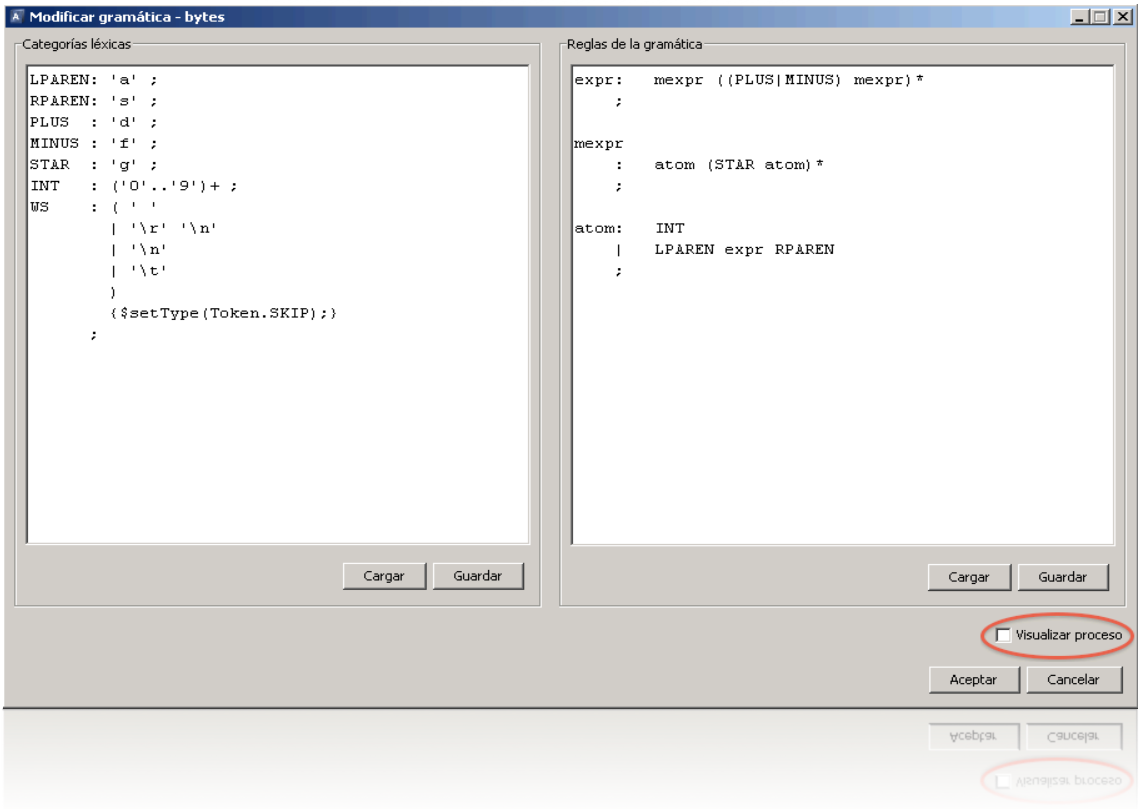
- Selección de **estilos para los comentarios** en el panel encargada de la configuración de los mismos:



- Edición sobre la tabla de delimitadores y supresión del botón de modificar para aplicar los cambios sobre la lista de los mismos:

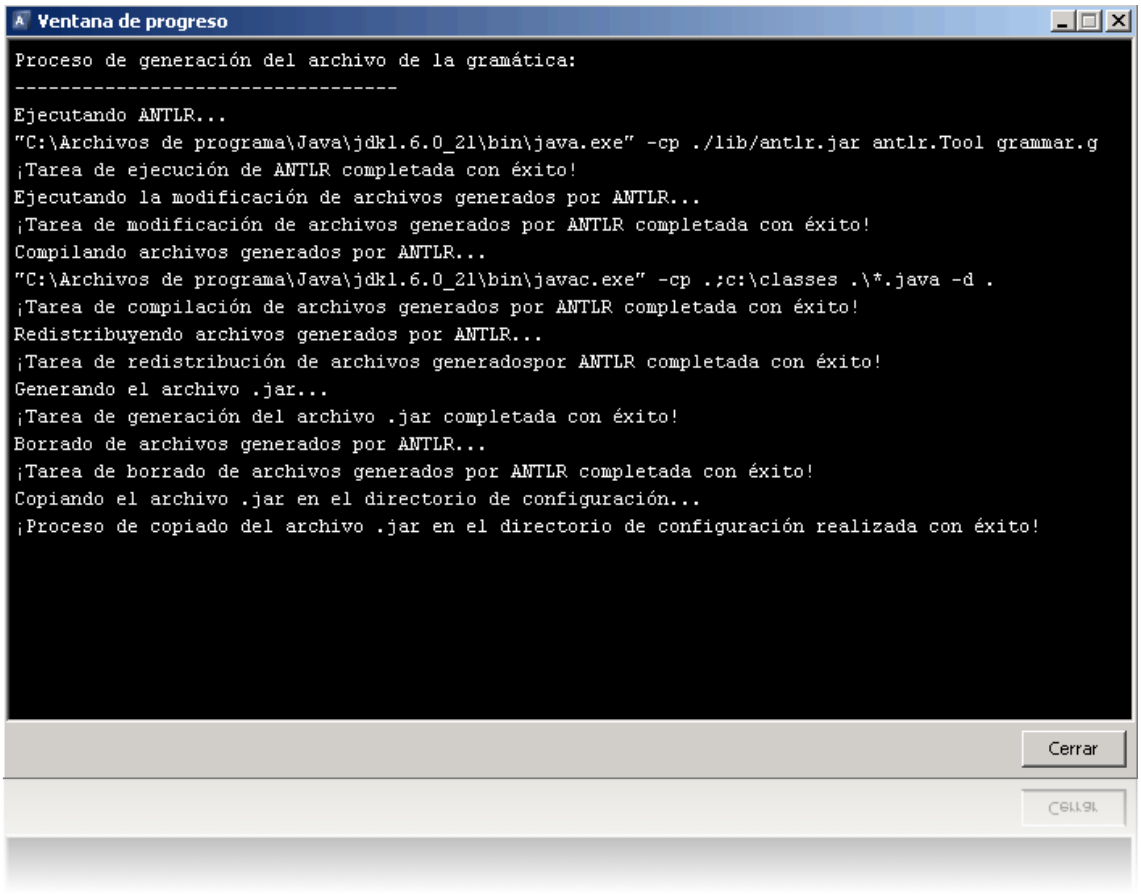
8.5.1.1 Ventana de Configuración de la Configuración Sintáctica

La ventana de configuración de la configuración sintáctica solamente presenta un novedad importante con respecto a las anteriores versiones:



Se trata del **JCheckBox** indicado por el círculo rojo en la anterior imagen que permite, si se selecciona el mismo, la **visualización del proceso de la generación del archivo de la gramática**.

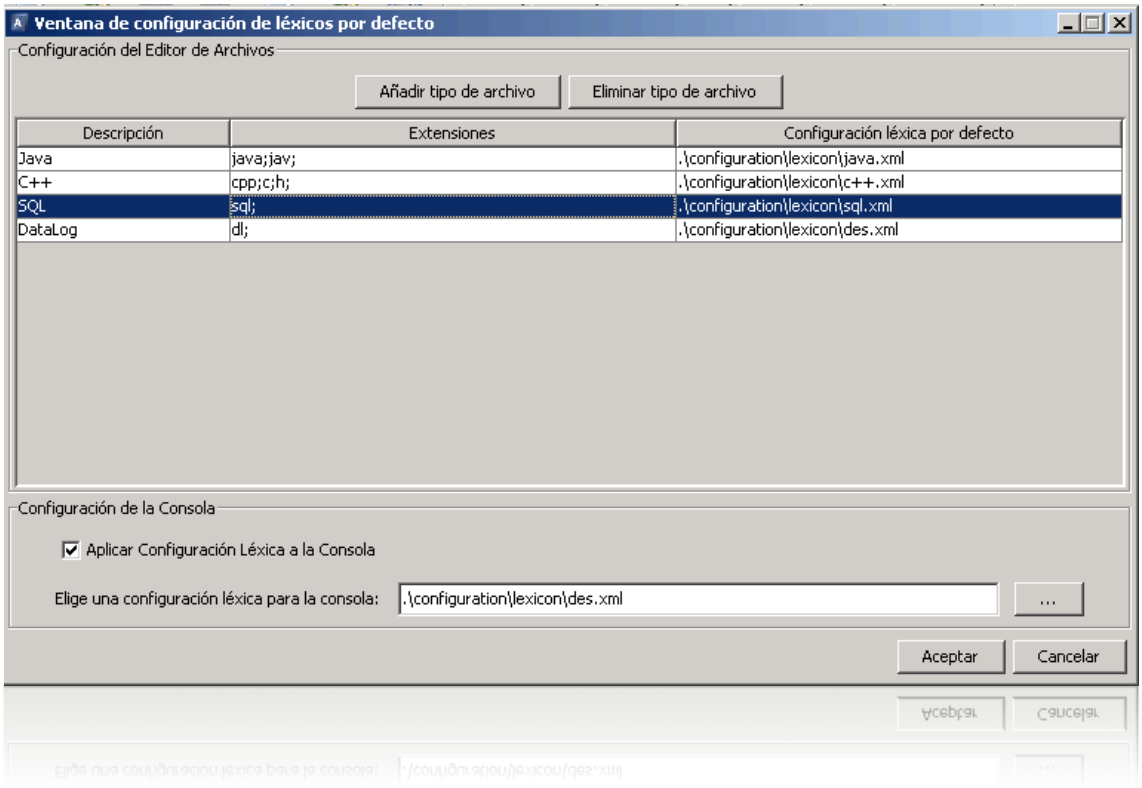
Ahora, si el usuario selecciona la opción correspondiente, se obtiene el siguiente resultado:



La ventana presenta todo el proceso de generación del archivo de la gramática paso por paso, habilitando el botón de **cerrar** una vez finalizado el proceso.

8.5.1.2 VENTANA DE CONFIGURACIÓN DE LÉXICOS POR DEFECTO

Ésta ventana ha sido añadida a la nueva versión de la aplicación y ofrece el siguiente aspecto:



La ventana presenta un panel superior de configuración de configuraciones léxicas para el *editor de archivos* con una tabla sobre la que el usuario podrá editar los valores de las configuraciones léxicas por defecto. Cada lista de extensiones pertenecientes a una configuración léxica está separada por *';'* y podrán ser añadidas o eliminadas con los botones **añadir tipo de archivo** y **eliminar tipo de archivo** respectivamente. Para elegir los archivos de configuración léxica a aplicar a cada grupo, el usuario podrá seleccionar el archivo mediante la *opción de menú contextual* en la columna de la tabla correspondiente.

El panel inferior muestra la configuración de configuraciones léxicas por defecto para el *panel de la consola* en el que el usuario tendrá la opción de elegir la configuración léxica a aplicar sobre el panel de la consola así como si realmente quiere que se aplique a la misma mediante el **JCheckBox** correspondiente.

8.6 Objetivos Cumplidos

Los objetivos que finalmente han sido cumplidos de una forma satisfactoria en éste proyecto han sido los siguientes:

8.6.1 General

- Limpieza y estandarización del código fuente de la aplicación.
- Gestión segura de hilos al iniciarse la aplicación.
- Aplicación de iconos a las opciones en la barra de menú.
- Aplicación de iconos a las opciones en los menús contextuales.
- Sincronización entre editor de archivos, panel del explorador de archivos, barra de estado, barra de herramientas y barra de menús en todo momento.
- Nueva ventana de Splash.
- Nueva imagen en ventana de About Us.

8.6.2 Editor de Archivos

- Cambio de iconos en el panel de pestañas.
- Asignación de configuración léxica a nivel de archivo y no a nivel de proyecto.
- Asignación de configuración sintáctica a nivel de archivo y no a nivel de proyecto.
- Marcado de paréntesis, corchetes y llaves arreglado en los archivos en el editor de archivos.
- Solucionados problemas con los archivos nuevos y con la pestaña de log y su guardado en la configuración de los proyectos y gestor de espacio de trabajo.
- Activación de *line wrapping* en los editores de texto.
- Desactivación de *line wrapping* en los editores de texto.
- Activación de sangrado automático en los editores de texto.
- Desactivación de sangrado automático en los editores de texto.
- Configuración del aspecto gráfico del editor de textos.

- Envío del contenido del archivo activo en el editor de archivos al panel de la consola.
- Definición de máximo número de líneas a enviar a la consola desde el editor de archivos.
- Las operaciones de deshacer y rehacer sitúan el foco en el archivo donde se ha producido el cambio.
- Cuando un usuario selecciona la pestaña en el editor de archivos, el foco se traslada automáticamente a la zona de edición del mismo.
- Scrolling línea a línea en el editor de texto mediante *rueda de ratón*.
- Scrolling línea a línea en el editor de texto mediante *CTRL+Flecha*.
- Habilitación de modo sobreescritura en los editores de texto mediante la pulsación de la tecla *INSERT*.
- Deshabilitación de modo sobreescritura en los editores de texto mediante la pulsación de la tecla *INSERT*.
- Adición de un nuevo componente para el mostrado de las líneas de los archivos en el editor de texto.
- Cargado de archivos en el editor de archivos sin línea en blanco adicional al final.
- Arreglado el comportamiento incorrecto de los archivos principales o compilables cuando están abiertos pero no pertenecen a un proyecto.
- *CTRL+Shift+F3* realiza búsquedas hacia atrás del texto seleccionado en el archivo activo en el editor de archivos.

8.6.3 CONFIGURACIÓN LÉXICA

- Separación de paneles en distintas pestañas en la ventana de configuración léxica.
- Guardado automático de la configuración léxica cuando el usuario selecciona el botón de aplicar los cambios en la ventana de configuración de la configuración léxica.
- Edición sobre la tabla de delimitadores y eliminación del botón de modificar para aplicar los cambios realizados.

- Confirmación al salir de la ventana con *ESC* para aplicar los cambios realizados.
- Configuración de configuraciones léxicas por defecto para los archivos en función de su extensión.
- Configuración de configuraciones léxicas por defecto para el panel de la consola del sistema.
- Opción de guardar como añadida en la ventana de configuración léxica.
- Borrado de archivos temporales de configuración léxica arreglado.

8.6.4 Panel de la Consola

- Asignación de configuración léxica al panel de la consola.
- Configuración del aspecto gráfico del panel de la consola.
- Búsquedas en el panel de la consola.
- Parseado de variables de ACIDE disponible también mediante teclado por consola y no solamente mediante barra de herramientas.
- Cierre de la consola cargada en el panel de la consola.
- Reinicio de la consola cargada en el panel de la consola.
- Borrado del buffer del panel de la consola.
- Guardado del contenido de la consola a un archivo externo.
- Los procesos relativos a los ejecutables cargados en el panel de la consola se matan cuando los mismos son cerrados, reiniciados ó bien la aplicación se cierra.
- Sustitución de la línea actual en la consola por el comando pulsado en el barra de herramientas de comandos ejecutados en el panel de la consola.
- Scrolling línea a línea en el panel de la consola mediante *rueda de ratón*.
- Scrolling línea a línea en el panel de la consola mediante *CTRL+Flecha*.
- Eliminación de líneas en blanco extras producidas por la pulsación de la tecla *ENTER*.
- *Edición* solamente disponible tras la marca del prompt de la misma, impidiendo al usuario borrar contenido ya cargado por la consola.

- *Buffer máximo de número de líneas* para mostrar en el panel de la consola.

8.6.5 Panel del Explorador de Archivos

- Cambio de iconos en el panel de explorador de archivos.
- Comprobación de nombres en la creación de carpetas en el panel de explorador de archivos para evitar la creación de carpetas con el mismo nombre y en el mismo nivel del árbol de directorios.

8.6.6 Gestión de Proyectos

- Apertura de todos los archivos existentes de un proyecto en el editor de texto mediante opción de menú dejando el foco en la primera línea del primero de ellos.
- Adición de todos los archivos abiertos en el editor de texto a los proyectos mediante opción de menú.
- Selección múltiple de archivos en la opción de apertura de los mismos.
- Selección múltiple de archivos en la opción de adición de los mismos a los proyectos.
- Visualización del resultado de la ejecución de proyectos.
- Adición de los parámetros de configuración del compilador a la configuración de los proyectos.
- Recordatorio de rutas para último archivo y proyecto abiertos para las ventanas de selección de archivos.

8.6.7 Barra de Herramientas

- Organización de barra de herramientas por secciones: basada en el menú, basada en comandos ejecutados en la consola y basada en aplicaciones ejecutadas fuera de ACIDE respectivamente.
- Definición de parámetros extra para los comandos de la barra de herramientas que se ejecutan en el panel de la consola: ninguno, texto, archivo ó directorio.

- Opción añadida a los parámetros de la barra de herramientas que se ejecutan en el panel de la consola para que se ejecute en la consola del Sistema Operativo ó en ACIDE.
- Edición sobre las tablas de la ventana de configuración de la barra de herramientas y eliminación del botón modificar
- Visualización de barra de herramientas cuando ésta es más grande que la ventana mediante botones de scroll.

8.6.8 Barra de Estados

- Adición del panel que muestra el número de líneas del archivo activo en el editor de archivos en la barra de estados de la aplicación.
- Adición del panel que muestra el modo de escritura en el editor de archivos en la barra de estados de la aplicación.
- Captura de la pulsación de teclas de LOCK y su posterior actualización en la barra de estados en toda la aplicación.

8.6.9 Configuración Sintáctica

- Refactorización del proceso de generación de archivo de gramática.
- Visualización del proceso de creación de archivo de gramática para la configuración sintáctica de archivos disponible mediante opción en la ventana de configuración correspondiente.

8.6.10 Barra de Menús

- Refactorización del código del mismo.
- Generación automática de la ventana de configuración del menú en función de las opciones ya existentes y mejor distribución de las opciones de la misma en diferentes pestañas.
- Gestión de proyectos recientes mediante opción en la barra de menús.
- Gestión de archivos recientes mediante opción en la barra de menús.

8.6.11 VENTANA DE BÚSQUEDAS Y REEMPLAZAMIENTOS

- Eliminación del mensaje de confirmación antes de un reemplazamiento general.
- Mensaje al término de un reemplazamiento general informando del número de reemplazamientos realizados en el mismo.

8.7 OBJETIVOS NO CUMPLIDOS

Al principio del desarrollo del proyecto se pactaron algunos objetivos para el proyecto que no han podido ser resueltos finalmente y que se detallan a continuación:

- Definición de una interfaz gráfica que permitiese la edición de **DES** al estilo de **Microsoft Access**.
- Definición e implementación del *análisis sintáctico* así como su aplicación en **ACIDE**.
- Definición y aplicación de una *barra de menús totalmente parametrizable* al estilo de la barra de la herramientas que permitiese añadir nuevas opciones de menú de una forma sencilla.
- Aplicación de hilos para:
    - Apertura de archivos en el editor de archivos.
    - Apertura de proyectos en la aplicación.
    - Aplicación del formato léxico en el editor de archivos al estilo de **Microsoft Word** cuando se aplica la autocorrección. Esto permitiría la opción de poder devolver el control al usuario mientras los documentos se está aplicando la configuración léxica correspondiente.
- Refactorización de la ventana de **Buscar/Reemplazar** así como su funcionamiento: Se intentó juntar las funcionalidades de ambas ventanas para simplificar el código así como el análisis de los métodos de búsqueda y reemplazo. Esto provocó una ralentización de las búsquedas y reemplazamientos por lo que finalmente y tal y como se ha

mencionado con anterioridad, se ha optado por hacer un *rollback* al código original.

- Opción de *respetar mayúsculas/minúsculas* en los reemplazos: por la profunda reestructuración que hubiera supuesto adaptar el método para los reemplazamientos generales y por la premura de tiempo a última hora, se ha decido posponer la misma para futuras versiones de la aplicación.

8.8 CONCLUSIONES

Desde el mismo momento en el que el autor del presente documento inició la estandarización y optimización del código fuente supo de las dificultades que dichas operaciones entrañaban de cara a la entrega de una versión estable y confiable de la aplicación.

Después de todas las nuevas funcionalidades y todos los cambios introducidos en la aplicación, se puede concluir que ahora el desarrollador de la misma tendrá acceso a un código estandarizado y claro, con todas las clases perfectamente localizadas en su paquete correspondiente y con una documentación más elaborada acerca del funcionamiento de la aplicación con respecto a las precedesoras versiones, además de un software fiable y confiable.

Aún queda mucho camino por recorrer y muchas posibles mejoras por introducir y corregir, ya que ésa es la gran virtud de éste proyecto, la de no tener límites.

Pese a todo en líneas generales se ha conseguido el objetivo propuesto por el alumno desde el primer momento: aprendizaja acerca de desarrollo de un IDE, limpieza del código fuente, adición de nuevas funcionaliades y corrección de antiguos errores así como la **obtención de una versión distribuíble** de la aplicación.

9 POSIBLES AMPLIACIONES

A Continuación se detalla un lista con las posibles mejoras para la aplicación, tanto a nivel de código fuente como de funcionalidades:

9.1 CÓDIGO FUENTE

A nivel de código fuente se identifican las siguientes funcionalidades:

- Definición e implementación del **AcideExceptionManager:** Ésta clase será la encargada de gestionar todas y cada una de las excepciones del sistema.
- Definición e implementación de una jerarquía de excepciones que serán controladas por el **AcideExceptionManager.**
- Redefinición del **AcideUndoManager** para que evite comportamientos indeseados en los editores de archivos tales como:
    - Paso extra al realizar el CTRL+Z ó CTRL+Y cuando se produce un reemplazamiento general.
    - Paso extra al realizar el CTRL+Z ó CTRL+Y cuando se está en el modo sobreescritura en el editor de texto.
    - Cuando se pulsa CTRL+Z o CTRL+Y sobre el editor de texto a veces se pierde el cursor en el mismo ya que se genera una excepción en el método *modelToView()* dado que el caret de texto se pierde.
- Redefinición del **AcideTabbedPaneUI** para que vuelva a funcionar *JTabbedPane.SCROLL_TAB_LAYOUT* sobre el mismo y así evitar que se muestren las pestañas de una forma un tanto desorganizada.
- Definir una forma para evitar que cuando se cierren todas las ventanas de configuración que dependen de otras, éstas se minimicen o se cierren también al estilo de *ventanas modales* como los cuadros de diálogo de la clase **JFileChooser**.
- Adaptar la tabla en la ventana de configuración de la configuración léxica para que se acepte la edición de las palabras reservadas sobre la

misma como ocurre con las de la ventana de configuración de la barra de herramientas.

- Definición del **AcideSearchConfiguration** y del **AcideSearchRecord** para guardar las configuraciones de cada búsqueda realizada y llevar el registro de cada una de las búsquedas anteriores.
- Definición de la ventana **AcideSearchConfigurationWindow** en la que el usuario podrá acceder al historial de búsquedas y ejecutar la búsqueda que él seleccione.
- Definición de la ventana **AcideHelpWindow** para mostrar el contenido de la ayuda en formato HTML. Opcionalmente implementación de un índice para la búsqueda de conceptos en la misma.
- Definición de un **JScrollPanel** para mostrar la *lista de proyectos* y *archivos recientes* ya que ahora cuando ambas listas son muy grandes y no caben en la ventana, la opción de vaciar la lista se pierde.
- Implementación de la **barra de herramientas basada en menús** así como su adición a la ventana de configuración de la barra de herramientas y a la gestión de la configuración de la misma. Actualmente se ha dejado preparado a propósito para su implementación pero no se ha podido llevar a cabo.

9.2 FUNCIONALIDADES

En el máximo nivel de orden de prioridad en la lista que a continuación se procede a detallar, se encontrarían las tareas correspondientes a los objetivos no cumplidos mencionados con anterioridad.

Después de las misms y por orden de prioridad se encontrarían las siguientes:

- Permitir el *envío de SIGINT a la consola* cargada en el panel de la consola para detener su ejecución.
- Permitir la definición de *diferentes zonas* en la barra de herramientas para agrupar a los comandos por *intención*, como por ejemplo carga, listado o flags.

- Opción para habilitar/deshabilitar el parseado de las variables de ACIDE como *$activeFile$* ó *$mainFile$*.
- Opción de personalizar la *presentación* del componente que muestra las *líneas de los archivos* en el editor de archivos.
- Permitir definir si el sangrado automático es con *espacios* ó con *tabulaciones*.
- Opción para habilitar y deshabilitar la adición de *cada línea* enviada desde el editor de archivos al panel de la consola como un *comando diferente* en el historial de comandos del mismo.
- Opción para la adición o no adición de *comillas* a los comandos enviados al panel de la consola o a la consola del sistema operativo.
- *Ventana de progreso* mientras duren los *reemplazamientos* y que vaya informando del transcurso de los mismos.
- Opción de *respetar las minúsculas/mayúsculas* en los reemplazamientos de la ventana de reemplazos.
- Opción para *añadir o eliminar marca de comentario* en el editor de archivos.
- Añadir un nuevo panel en la ventana de configuración de configuraciones léxicas para la configuración de los pares de delimitadores.
- Permitir la aplicación de comentarios multilínea.
- Cuando una configuración léxica no tiene definida ningún delimitador, la aplicación del estilo de las palabras reservadas no se aplica.
- Manipulación de *palabras reservadas* de una configuración léxica mediante la *edición directa sobre la tabla* en la ventana de configuración de configuraciones léxicas.
- Permitir localizar la aplicación a *otros idiomas* y adición del nuevo idioma como opción disponible en la barra de menús.
- Permitir la *multiselección* de archivos en el árbol del explorador de proyectos.

- *Mostrar ordenados alfabéticamente* primero las carpetas y después los archivos del proyecto en el árbol del explorador de archivos.
- Permitir *Drag and Drop* en el árbol de explorador de archivos para permitir la ordenación de los mismos de una forma manual.
- *Asociación de archivos del sistema operativo* a ACIDE.
- Permitir la ejecución de *múltiples instancias* de ACIDE a la vez.
- Permitir *Drag and Drop* para archivos desde aplicaciones del sistema operativo a ACIDE con objeto de poder abrirlo en un editor de ACIDE ó añadirlo a los proyectos.
- Definición de una extensión para los archivos de configuración léxica.
- Organización de los archivos en el editor de archivos mediante *Cascade/Tile Horizontal/Tile Vertical*.
- *División vertical* de los archivos en el editor de archivos, permitiendo la división en *4 partes diferentes*.
- Modificación del tipo de fuente del léxico por grupos para su posterior aplicación a categorías sintácticas diferentes, como por ejemplo los tipos, funciones, etc.
- Definición de *diccionario de idioma*.
- Implementación de *macros* de comandos.
- Obtención automática de la configuración léxica a partir de la configuración sintáctica.
- *Comprobación de tipos al vuelo*.

10 LISTA DE PALABRAS CLAVE

- Análisis léxico
- Análisis sintáctico
- Entorno de desarrollo integrado (IDE)
- Consola
- Gestión de proyectos
- Editor de código
- Lenguajes de programación compilados e interpretados
- Edición multi-archivo
- Coloreado de palabras
- Configurable

11 BIBLIOGRAFÍA

Para el desarrollo del proyecto se han consultado las siguientes fuentes:

- **Line wrapping en JTextPane:**
  - *http://objectmix.com/java/73385-disabling-word-wrap-jtextpane.html*
- **Sangrado automático:**
  - *http://www.thatisjava.com/java-core-gui-apis/42873/*
- **Modo de sobreescritura en JTextPane:**
  - *http://www.thatisjava.com/java-core-gui-apis/50142/*
- **Barra de herramientas con botones de scroll:**
  - *http://www.progdoc.de/papers/ScrollableBar/ScrollableBar.html*
  - *http://www.java.net/external?url=http://tech.chitgoks.com/2009/12/05/create-a-sideway-scrolling-toolbar-using-java/*
- **Componentes de Swing en general:**
  - *http://download.oracle.com/javase/tutorial/uiswing/components/index.html*
- **Ejemplos de código y tutoriales:**
  - *http://www.java2s.com*
  - *www.javabeginner.com*
- **Aplicación de la configuración léxica:**
  - *http://www.linuxquestions.org/questions/programming-9/java-jtextarea-and-syntax-highlighting-417575/*
- **SwingWorker:**
  - *http://java.sun.com/products/jfc/tsc/articles/threads/threads2.html*
- **Threads en java:**
  - *http://download.oracle.com/javase/tutorial/essential/concurrency/*

- **CTRL+Rueda ratón para hacer el zoom:**
  - *http://www.icefaces.org/JForum/posts/list/0/16103.page*
- **Ejecución de una sola instancia de aplicación en java:**
  - *http://es.debugmodeon.com/articulo/ejecutar-solo-una-instancia-de-una-aplicacion*
- **JScrollPane con flechas:**
  - *http://www.java-forums.org/awt-swing/11387-jscrollpane-arrowkeys.html*
- **Cascade and Tile:**
  - *http://www.thatsjava.com/java-swing/4349/*
- **Lectura y escritura de ficheros de texto en java:**
  - *http://chuwiki.chuidiang.org/index.php?title=Lectura_y_Escritura_de_Ficheros_en_Java*

12 REFERENCIAS

[1] D. Cardiel Freire, J. J. Ortiz Sánchez y D. Rupérez Cañas. ACIDE: A Configurable IDE. Universidad Complutense de Madrid. 2007.

[2] Página oficial del editor de textos JEdit. *http://www.jedit.org/*

[3] Pagina oficial del editor WinEdt. *http://www.winedt.com/*

[4] Página oficial del editor Crimson Editor. *http://www.crimsoneditor.com/*

[5] Página oficial del IDE Eclipse. *http://www.eclipse.org/*

[6] Página oficial del IDE Netbeans. *http://netbeans.org/*

[7] Página oficial del cliente subversion. *http://subversion.tigris.org/*

[8] Página oficial de Google Code. *http://code.google.com/*

[9] Página oficial de DES. *http://www.fdi.ucm.es/profesor/fernan/des/html/contributions.html*

[10] Página oficial de Sun. *http://www.java.com/es/download/manual.jsp*

13 INFORMACIÓN DE CONTACTO

Dado que es código libre tenemos todo el código fuente y el ejecutable en las siguientes direcciones de internet:

- **Ejecutable:** *http://acide.sourceforge.net*
- **Código fuente:** *http://code.google.com/p/acide-0-8-release-2010-2011/*

Para ponerse en contacto con el desarrollador del proyecto se puede usar la siguiente dirección de correo electrónico:

- **Javier Salcedo Gómez:** *salcedonia@gmail.com*

14 AUTORIZACIÓN

Se autoriza a la Universidad Complutense a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, la documentación y/o el prototipo desarrollado.

Javier Salcedo Gómez